Задачи и упражнения по программированию

2

ТЯЖЕЛАЯ ПРОМЫШЛЕННОСТЬ И ТРАНСПОРТ

«ВЫСШАЯ ШКОЛА»

Задачи и упражнения по программированию

2 КНИГА

ТЯЖЕЛАЯ ПРОМЫШЛЕННОСТЬ И ТРАНСПОРТ

Издание 2-е, дополненное

Под редакцией
доктора технических наук,
профессора
А. Я. САВЕЛЬЕВА

Москва «Высшая школа» 1989

ББК 22.18
З-15
УДК 681.3.06

Рекомендовано к изданию
Государственным комитетом СССР
по народному образованию

В.Е. Алексеев, А.С. Ваулин

Рецензенты: канд. техн. наук В.В. Смирнов (МИСИ),
А.А. Цыганов (ПТУ № 129, Москва)

Задачи и упражнения по программированию: В 5-ти кн.: Практ.
З-15 пособие для ПТУ /Под ред. А.Я. Савельева. Кн. 2. Тяжелая промышленность и транспорт/В.Е. Алексеев, А.С. Ваулин. – 2-е изд., доп. – М.: Высш. шк., 1989. – 112 с.: ил.
ISBN 5–06–000310–8

Книга содержит материал для практического изучения приемов разработки алгоритмов типовых задач по программированию на языке БЕЙСИК алгоритмов различных структур с использованием характерных приемов программирования и подпрограмм.

Второе издание (1-е – в 1986 г. "Сборник задач и упражнений по программированию") дополнено сведениями о построении графиков, таблиц и гистограмм.

Для учащихся профтехучилищ. Может быть использовано широким кругом читателей, желающих овладеть приемами программирования.

З $\frac{2404010000\,(4307000000)-153}{052\,(01)-89}$ 66–89

ББК 22.18

517.8

ISBN 5–06–000310–8

ПРЕДИСЛОВИЕ

В Основных направлениях экономического и социального развития СССР на 1986–1990 годы и на период до 2000 года подчеркивается, что необходимо высокими темпами наращивать масштабы применения современных электронно-вычислительных машин всех классов.

Развитие науки и техники постоянно ставит задачу дальнейшего совершенствования подготовки современного рабочего, которая позволила бы ему хорошо ориентироваться в системе данного производства, свободно переходить на смежные по технологии участки работы, совмещать функции эксплуатации и обслуживания. Следовательно, в подготовке рабочего наряду с профессиональным мастерством большое значение приобретает овладение основами науки, понимание научных принципов развития техники, технологии и организации производства.

Научно-технический прогресс в области микропроцессорной техники привел к созданию микроЭВМ, которые в настоящее время широко применяются в различных отраслях народного хозяйства.

Высокая производительность и надежность, малые габаритные размеры, низкое электропотребление и невысокая стоимость микропроцессоров, обладающих большими функционально-логическими возможностями, позволили разработать ряд специализированных и универсальных ЭВМ. Специализированные ЭВМ применяются в управлении оборудованием и технологическими процессами, в создании широкого класса цифровых вычислительных средств и средств цифровой автоматики. Универсальные ЭВМ широко используются при проведении расчетов, моделировании и автоматизации проектирования.

В связи с этим появляется необходимость подготовки рабочих кадров, сочетающих профессиональные знания с умением и навыками практического использования ЭВМ. Введенный курс "Основы информатики и вычислительной техники" является одной из дисциплин общетехнической подготовки учащихся. Она должна способствовать увеличению объема расчетных работ в общеобразовательных, общетехнических и специальных дисциплинах, углублению понимания сущности физических и химических явлений в технологических процессах и их закономерностей.

Особенностью сборника задач является то, что он предназначен для изучения программирования на языке БЕЙСИК и использования его при решении профессиональных задач.

Язык БЕЙСИК является наиболее распространенным языком микроЭВМ, в том числе ЭВМ типа "Искра-226", ДВК-1, ДВК-2, ДЗ-28 и других, использующихся в училищах профтехобразования.

Одним из видов подготовки рабочих различных профессий являются практические занятия. Помимо приобретения практических навыков они должны развивать обобщенное мышление, так как без этого невозможно научить будущих специалистов самым разнообразным приемам.

Подготовка задач к решению на ЭВМ во многом способствует развитию абстрактного мышления, связанного с формализацией задач, разработкой алгоритмов и программ. Поэтому наряду с профессиональными задачами в сборнике задач отражены приемы программирования типовых алгоритмов, которые могут быть использованы в решении практических задач различных профессий.

Пособие состоит из двух глав, ответов и приложения. В гл. 1 рассмотрены этапы подготовки задач для решения на ЭВМ. Особое внимание уделено этапу алгоритмизации задач по профессиям тяжелой промышленности и транспорта; в гл. 2 рассмотрено программирование на языке БЕЙСИК на примерах типовых задач по этим профессиям.

В приложениях даны рекомендации по практическому использованию микроЭВМ типа ДВК и "Искра-226" и вывод результатов в виде графиков, таблиц, гистограмм.

Задачи, отмеченные символом ●, имеют ответ, а символом ▲ являются задачами повышенной трудности.

Авторы выражают благодарность коллективу кафедры АСУ Московского инженерно-строительного института им. В.В. Куйбышева, канд. техн. наук, доц. В.В. Смирнову, старшему мастеру СПТУ № 129 А.А. Цыганову, взявшим на себя труд по рецензированию настоящего пособия и сделавшим ряд ценных замечаний.

Авторы

ГЛАВА ПЕРВАЯ

АЛГОРИТМИЗАЦИЯ ЗАДАЧ

§ 1. ПОДГОТОВКА ЗАДАЧ ДЛЯ РЕШЕНИЯ НА ЭВМ

Процесс подготовки и решения задач на ЭВМ является пока достаточно сложным и трудоемким, требующим выполнения целого ряда этапов.

▷ **Такими этапами являются:**

1) *постановка задачи;*
2) *математическая формулировка задачи;*
3) *выбор численного метода решения;*
4) *разработка алгоритма решения задачи;*
5) *написание программы;*
6) *ввод программы и исходных данных;*
7) *отладка программы;*
8) *решение задачи на ЭВМ.*

Данная последовательность характерна для решения каждой задачи. Однако в процессе подготовки задачи каждый этап может иметь более или менее выраженный характер. Выполнение этапов в процессе подготовки задачи носит характер последовательного приближения, так как уточнение задачи на последующем этапе приводит к необходимости возврата к предыдущему и повторному выполнению последующих этапов.

Рассмотрим подробнее выполнение работ на каждом этапе в процессе подготовки задачи к решению.

Постановка задачи определяет цель решения задачи, раскрывает ее содержание. Задача формулируется на уровне профессиональных понятий, должна быть корректной и понятной исполнителю (пользователю). Ошибка в постановке задачи, обнаруженная на последующих этапах, приведет к тому, что работа по подготовке задачи к решению должна начаться с самого начала.

Математическая формулировка задачи осуществляет формализацию задачи путем описания ее с помощью формул, определяет перечень исходных данных и получаемых результатов, начальные условия, точность вычисления. По существу разрабатывается математическая модель решаемой задачи.

Выбор численного метода решения. В ряде случаев одна и та же задача может быть решена с помощью различных численных методов. Выбор метода должен определяться многими факторами, основными из которых являются точность результатов решения, время решения на ЭВМ и объем оперативной памяти. В каждом конкретном случае в качестве критерия для выбора численного метода принимают какой-либо из указанных критериев или некоторый интегральный критерий.

В простых задачах данный этап может отсутствовать, так как сам численный метод определен математической формулировкой задачи. Например, вычисление площади треугольника по формуле Герона, корней квадратного уравнения и др.

Разработка алгоритма решения задачи. На данном этапе устанавливается необходимая логическая последовательность вычислений с учетом выбранного численного метода решения и других действий, с помощью которых будут получены результаты.

Алгоритм – некоторая конечная последовательность предписаний (правил), определяющая процесс преобразования исходных и промежуточных данных в результат решения задачи.

Таким образом, при разработке алгоритма решения задачи математическая формулировка задачи является основой для определения последовательности действий, приводящих к получению искомого результата.

▷ **Разрабатываемый алгоритм должен обладать следующими свойствами:**

массовостью, позволяющей решать не одну задачу, а целый класс задач;

детерминированностью, однозначно определяющей выполняемые действия (промежуточные и окончательные результаты разных пользователей будут одинаковыми при одинаковых исходных данных);

результативностью, позволяющей получить результат после конечного числа шагов.

По используемой структуре управления вычислительным процессом алгоритмы классифицируют следующим образом: линейной структуры; разветвляющейся структуры; циклической структуры; со структурой вложенных циклов; смешанной (комбинированной) структуры.

Для иллюстрации алгоритмов любой структуры используются простые математические формулировки задач, доступные учащимся любых профессий. Для решения таких задач во многих случаях может оказаться нецелесообразным использование ЭВМ, однако рассмотрение способов их программирования имеет смысл, так как они являются составной частью более сложных задач.

При решении любой более или менее сложной задачи могут иметь место несколько различных алгоритмов, приводящих к получению результата. Из всех возможных алгоритмов следует выбирать наилучший в смысле некоторого критерия.

Алгоритм линейной структуры – алгоритм, в котором все действия выполняются последовательно друг за другом. Такой порядок выполнения действий называется естественным (задачи № 1–9).

На практике редко удается представить схему алгоритма решения задачи в виде линейной структуры, так как задачи содержат различные условия или требуют многократного повторения вычислений.

Алгоритм разветвляющейся структуры – алгоритм, в котором в зависимости

от выполнения некоторого логического условия вычислительный процесс должен идти по одной или другой ветви.

В общем случае количество ветвей в алгоритме разветвляющейся структуры может быть больше двух (задачи № 10–15).

Алгоритм циклической структуры – алгоритм, содержащий многократно выполняемые участки вычислительного процесса, называемые циклами.

Использование циклов позволяет существенно сократить схему алгоритма. Различают циклы с заданным и неизвестным числом повторений, характеризующиеся последовательным приближением к исходному значению с заданной точностью (задачи № 16–36, 38, 39).

Алгоритм со структурой вложенных циклов – алгоритм, содержащий цикл, внутри которого размещены один или несколько других циклов.

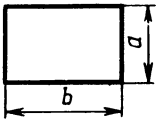
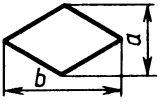
Цикл, охватывающий внутренние циклы, называется *внешним*. Правила организации как *внешнего*, так и *внутренних* циклов те же, как и для обычного цикла. Параметры этих циклов изменяются не одновременно, т.е. при одном значении параметра внешнего цикла параметр внутреннего цикла принимает по очереди все значения (задача № 37).

Существует много способов записи алгоритмов, отличающихся друг от друга наглядностью, компактностью, степенью формализации и другими показателями. Наибольшее распространение получили графический способ и так называемый алгоритмический язык записи алгоритмов, ориентированный на человека (псевдокоды).

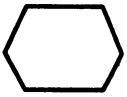
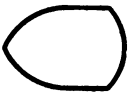
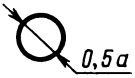
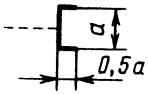
Графическая запись алгоритма должна выполняться в соответствии с государственными стандартами (ГОСТ 19.002–80 "Схемы алгоритмов и программ. Правила выполнения"; ГОСТ 19.003–80 "Схемы алгоритмов и программ. Обозначения условные и графические").

Схема алгоритма представляет собой последовательность блоков, предписывающих выполнение определенных действий, и связи между ними. Символы, наиболее часто используемые в сборнике задач, приведены в табл. 1.

Таблица 1

Наименование	Обозначение	Функции
Процесс		Вычислительное действие или последовательность вычислительных действий
Решение		Проверка условия и выбор направления выполнения алгоритма

Продолжение табл. 1

Наименование	Обозначение	Функции
Модификация		Начало цикла
Предопределенный процесс		Вычисление по подпрограмме
Дисплей		Ввод–вывод данных
Пуск–останов		Начало, конец обработки данных
Соединитель		Указание связи между прерванными линиями связи
Комментарий		Связь между элементами схемы и пояснениями

Примечание. Размер a выбирается из ряда 10, 15, 20 мм; $b = 2a$.

Выделение составных частей алгоритма должно определяться внутренней логикой процесса вычислений.

Схема алгоритма может выполняться с разной степенью детализации. Схема, в которой определены ввод и вывод информации и учитываются особенности языка программирования, называется *схемой программы.*

Запись алгоритма на алгоритмическом языке, ориентированном на человека, выполняется с помощью служебных слов и команд, которые записываются в сокращенном виде и подчеркиваются. Запись начинается со служебного слова алгоритм (АЛГ), за которым записывается его краткое название и определяются типы используемых величин. Далее перечисляются аргументы (АРГ) и результаты (РЕЗ). Команды, определяющие действия, записываются между служебными словами начало (НАЧ) и конец (КОН). Команды управления ходом вычислений начинаются служебными словами: ЕСЛИ, ТО, ИНАЧЕ, ЦК (цикл), КЦ (конец цикла), ПОКА. Команды друг от друга отделяются точкой с запятой.

> Общий вид записи алгоритма на алгоритмическом языке выглядит следующим образом:

АЛГ название алгоритма;
АРГ . . . ; **РЕЗ** . . . ;
НАЧ
Последовательность команд
КОН

> Команда разветвления, содержащая условие, имеет следующий вид записи:

ЕСЛИ условие
ТО последовательность команд
ИНАЧЕ последовательность команд
ВСЕ

> Команда цикла имеет следующий вид:

ПОКА условие
ЦК
Последовательность команд
КЦ

Объектами действий в алгоритмах являются числа, простые переменные и переменные с индексами (элементы массивов).

Массив – упорядоченная последовательность значений, имеющих одно имя.

В процессе решения простая переменная может изменять свои значения, но в каждый момент времени известно (хранится в памяти ЭВМ) только одно "текущее" значение. Простую переменную обозначают ее символическим именем (идентификатором). Элемент массива (переменная с индексом) состоит из имени и индексов, указывающих на расположение элемента в массиве.

Написание программы осуществляется по разработанному алгоритму с помощью языка программирования.

Программа представляет собой последовательность операторов языка, записанных в соответствии со схемой программы.

Ввод программы и исходных данных выполняется с помощью клавиатуры ЭВМ.

Основные инструкции по работе с ЭВМ даны в приложении.

Отладка программ представляет собой процесс обнаружения и устранения синтаксических и логических ошибок.

Синтаксические ошибки, связанные с неправильной записью конструкций языка, выявляются самой ЭВМ. Логические ошибки появляются в результате нарушения последовательности вычислений, правильности написания выражений и отсутствия необходимых данных для ЭВМ. Для выявления логических ошибок используют контрольные вычисления,

выполняемые на микрокалькуляторах, предусматривают в программе вывод промежуточных результатов и используют отладочные операторы. После отладки программы и проверки ее на тестовых данных дополнительные операторы, введенные для отладки, исключаются из программы.

Решение задачи на микроЭВМ обычно проводится в диалоговом режиме. В этом режиме пользователь с помощью клавиатуры ЭВМ может осуществлять ввод программы и ее корректировку, трансляцию программы (перевод с языка программирования на машинный), исправление синтаксических и логических ошибок при отладке, получение на выходе результатов и вспомогательной информации, необходимой для управления работой ЭВМ.

В последнее время встречается понятие составления общего плана решения, включающее выбор численного метода и разработку алгоритма.

Большое разнообразие задач и их профессиональная специфика присутствуют только на этапе постановки задачи. В дальнейшем, после математической формулировки выбора численного метода решения задачи получается некоторая математическая модель. Такая модель является абстрактной и может описывать процессы разной физической сущности.

Решение большинства практических задач требует использования некоторого набора характерных приемов алгоритмизации задач, таких, как организация цикла с несколькими одновременно изменяющимися параметрами, запоминание результатов, накопление суммы и произведения, вычисление многочлена, нахождение наибольшего и наименьшего значения.

Из этих и других приемов, как из элементарных "кирпичиков", строятся схемы алгоритмов решения задач любой сложности.

§ 2. ПРИМЕРЫ СОСТАВЛЕНИЯ АЛГОРИТМОВ

Алгоритмы линейной структуры

П. 1. *Математическая формулировка задачи*

Определить площадь треугольника по формуле Герона $s = \sqrt{p(p-a)(p-b)(p-c)}$, где a, b, c – длины сторон; $p = (a+b+c)/2$ – полупериметр треугольника.

Алгоритм решения задачи. 1. Ввод исходных данных a, b, c. 2. Вычисление p. 3. Вычисление s. 4. Вывод результата s.

Схема программы, описывающей алгоритм, представлена на рис. 1.

В приведенной схеме программы сначала вычисляется значение полупериметра p, а затем значение s, при вычислении которого используется p. Это позволяет избежать повторения вычислений одной и той же величины, а следовательно, уменьшить время решения задачи. На схеме блоки расположены в той последовательности, в которой они должны выполняться. Любая перестановка блоков приведет к невозможности решения задачи.

П. 2. *Математическая формулировка задачи*

Вычислить поверхность усеченного конуса $S = \pi(R+r)l + \pi R^2 + \pi r^2$ и его объем $V = 1/3\,\pi(R^2 + r^2 + Rr)h$.

Алгоритм решения задачи. 1. Ввод исходных данных R, r, l, h. 2. Вычисление S. 3. Вычисление V. 4. Вывод результатов S и V.

Схема программы, описывающая алгоритм, аналогична схеме, представленной на рис. 1.

Алгоритмы разветвляющейся структуры

П. 3. *Математическая формулировка задачи*

Вычислить значение функции $z = 1/(xy)$.

Казалось бы, что решение этой задачи можно описать алгоритмом линейной структуры. Однако для удовлетворения свойств массовости и результативности алгоритма необходимо, чтобы при любых исходных данных (значениях x и y) был получен результат или сообщение о том, что задача не может быть решена при заданных значениях исходных данных. Действительно, если $x = 0$ или $y = 0$, то задача не может быть решена, так как деление на нуль невозможно. Поэтому в алгоритме необходимо предусмотреть вывод информации для случая, когда вычисление z невозможно. Такой вычислительный процесс можно описать следующим выражением:

вычислить $z = 1/xy$, если $xy \neq 0$ (ветвь 1);

вывести $xy = 0$, если $xy = 0$ (ветвь 2).

Алгоритм решения задачи

1. Ввод исходных данных x, y. 2. Проверка условия $xy = 0$; если оно выполняется, то вывод $xy = 0$, в противном случае вычисление $z = 1/(xy)$. 3. Вывод результата z.

Схема программы представлена на рис. 2.

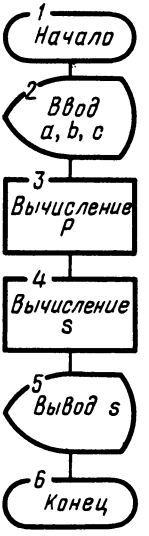


Рис. 1

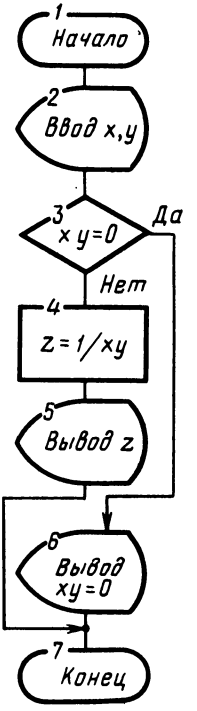


Рис. 2

После условного блока располагаются сначала блоки первой ветви (*4, 5*), а затем второй ветви (блок *6*). Поскольку после выполнения первой ветви нет надобности выполнять блок *6*, осуществляется переход сразу к концу алгоритма (блоку *7*).

Алгоритмы циклической структуры

П. 4. *Математическая формулировка задачи*

Вычислить значения функции $z = \sin x/x$ при x, принимающем значения 0,1; 0,2; ...; 1, т. е. изменяющемся от 0,1 до 1 с шагом 0,1.

Алгоритм решения этой задачи требует десять раз вычислять и выводить значения $z = \sin x/x$ при различных значениях аргумента x. Это цикл с заданным числом повторений, которое определяется как $n = [\frac{x_{кон} - x_{нач}}{h}] + 1$, где $x_{кон}$, $x_{нач}$ – конечное и начальное значения аргумента (параметра цикла); h – шаг его изменения. Прямоугольные скобки означают целую часть от деления.

Алгоритм решения задачи.

1. Задание перед циклом начального значения параметра цикла $x = 0{,}1$. 2. Вычисление z. 3. Вывод z. 4. Изменение x путем прибавления к нему шага изменения параметра $h = 0{,}1$. 5. Проверка условия окончания цикла $x > 1$; если оно не выполняется, то переход к началу цикла (блок *3*), в противном случае выход из цикла.

Схема программы, описывающая алгоритм, представлена на рис. 3.

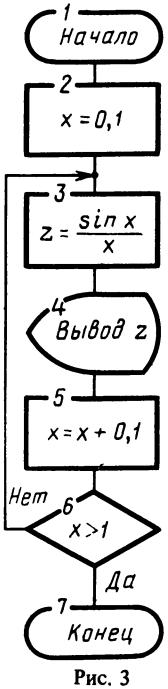


Рис. 3

Блок *2* задает начальное значение параметра цикла $x = 0{,}1$, блок *5* изменяет параметр цикла, блок *6* проверяет условие окончания цикла и управляет им, т.е. осуществляет переход к началу цикла, если он не закончен, или прекращает вычисление. Все эти действия необходимы для организации любого цикла. Схема прог-

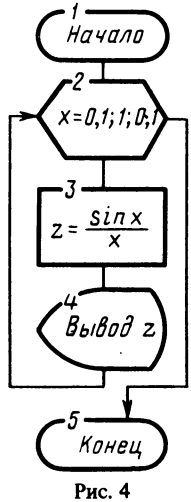


Рис. 4

раммы во многих случаях становится более компактной и наглядной, если для ее построения использовать блок начала цикла, который выполняет все функции, необходимые для организации цикла (рис. 4).

П. 5. *Математическая формулировка задачи*

Определить значение k, при котором $x^k/k \leqslant \epsilon$, где $k = 1, 2, 3, ...$

Для решения этой задачи необходимо организовать цикл, в котором изменяется параметр k от 1 с шагом 1. Это цикл с неизвестным числом повторений, так как результатом решения задачи является значение параметра k, для которого выполняется поставленное условие.

Алгоритм решения задачи.

1. Ввод исходных данных x, ϵ. 2. Задание начального значения параметра цикла $k = 1$. 3. Проверка условия $x^k/k \leqslant \epsilon$; если оно не выполняется, то производится изменение параметра цикла на единицу и переход к началу цикла (проверка условия), в противном случае выход из цикла. 4. Вывод результата k.

Схема программы, описывающей алгоритм, представлена на рис. 5.

Алгоритм со структурой вложенных циклов

П. 6. *Математическая формулировка задачи*

Вычислить значения матрицы $z_{ij} = x_i y_j$, где $i = 1, 2, ..., 8$; $j = 1, 2, ..., 10$.

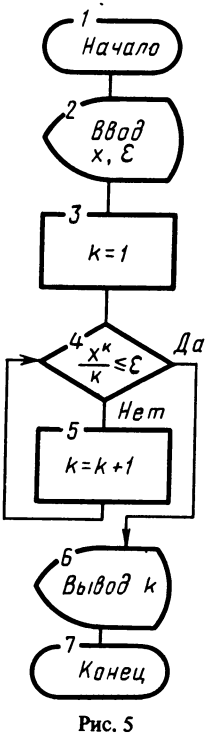


Рис. 5

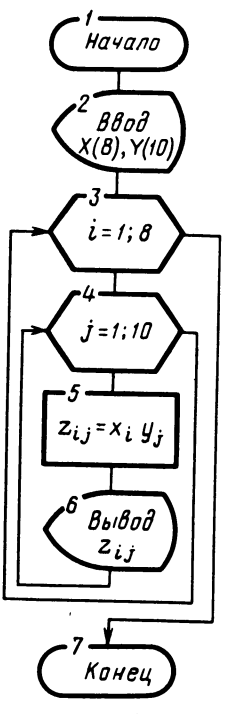


Рис. 6

Решение этой задачи требует вычисления всех возможных произведений x_i на y_j: взяв первое значение $x = x_1$, последовательно перемножить его на все значения y_j : x_1y_1, x_1y_2, ... , x_1y_{10}. Затем изменить значение $x = x_2$ и повторить вычисления, умножая на все значения y_j. Изменения x производить до последнего значения x_8. В результате вычислений будет получена матрица z, в которой индекс i определяет номер строки, а j – номер столбца. Изменение индекса i организовать во внешнем, а j во внутреннем цикле.

Алгоритм решения задачи.

1. Ввод значений элементов массивов x и y. 2. Организация цикла по i от 1 до 8. 3. Организация цикла по j от 1 до 10. 4. Вычисление z_{ij}. 5. Вывод z_{ij}. 6. Конец внутреннего цикла по j. 7. Конец внешнего цикла по i.

Схема программы, описывающая алгоритм, представлена на рис. 6. Если шаг изменения параметра цикла равен единице, то он на схеме не указывается.

Характерные приемы алгоритмизации задач

П.7. Организация цикла с несколькими одновременно изменяющимися параметрами.

Математическая формулировка задачи

Вычислить $z = a_ib_i$, где a изменяется от 1 до 3 с шагом 0,2, а b_i являются элементами массива $b_1, b_2, ..., b_{11}$.

Когда в цикле изменяются несколько параметров, закон изменения одного параметра должен задаваться в блоке начала цикла. Для изменения остальных параметров перед циклом необходимо задать их начальные значения, а внутри цикла вычислять текущие значения.

Для элемента массива параметром, изменяющимся в цикле, является его индекс. Следовательно, в этом цикле одновременно изменяются два параметра: переменная a от 1 до 3 с шагом 0,2; индекс от 1 до 11 с шагом 1. Закон изменения одного параметра, например индекса, задается в блоке начала цикла, тогда для изменения другого параметра необходимо перед циклом задать его начальное значение $a = 1$, а внутри цикла вычислять текущее, т. е. увеличивая его на шаг изменения 0,1 ($a = a + 0{,}1$).

Алгоритм решения задачи

1. Ввод значений элементов массива B. 2. Задание начального значения параметра $a = 1$. 3. Организация цикла по i от 1 до 11 с шагом 1. 4. Вычисление z. 5. Вывод z. 6. Изменение параметра a. 7. Конец цикла.

Схема программы, описывающая алгоритм, представлена на рис. 7.

П. 8. Запоминание результатов

Математическая формулировка задачи

Вычислить и запомнить $z_i = \sin ix_i/i$, где $i = 1, 2, ..., 10$.

В задачах, в которых требуется запомнить (сохранить) все вычисляемые значения, необходимо результаты записывать в массив значений. Запись в массив осуществляется в цикле, в котором изменяется индекс элемента массива.

При первом выполнении цикла, когда $i = 1$, будет вычислен и записан первый элемент массива z_1, при втором выполнении цикла – z_2 и далее до z_{10}.

Алгоритм решения задачи

1. Ввод значений массива x. 2. Организация цикла с параметром i, изменяющимся от 1 до 10 с шагом 1. 3. Вычисление в цикле элемента массива z_i и его запоминание. 4. Конец цикла. 5. Вывод всех значений массива Z. Схема программы, описывающая алгоритм решения задачи, представлена на рис. 8. Схему алгоритма можно изменить, разместив блок вывода элемента массива внутри цикла.

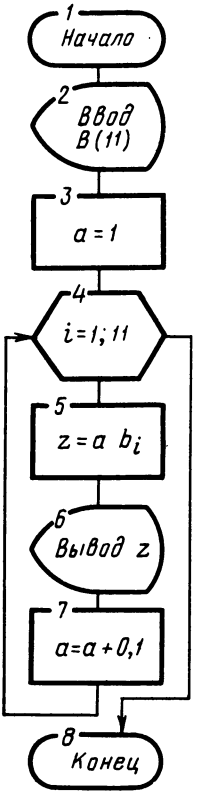


Рис. 7

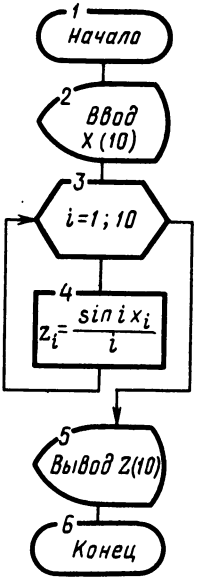


Рис. 8

П.9. Накопление суммы

Математическая формулировка задачи

Вычислить $z = \sum_{i=1}^{20} \frac{x_i}{i}$, где x_i – элементы массива $x_1, x_2, ..., x_{20}$.

Если необходимо накопить сумму значений некоторой функции при различных значениях аргумента, целесообразно организовать цикл, в котором необходимо не только вычислять значения функции, но и накапливать сумму путем прибавления полученного значения к сумме всех предыдущих слагаемых. Формула, используемая для накопления суммы, $z_i = z_{i-1} + y_i$, где y_i – слагаемое; z_{i-1}, z_i – предыдущая и последующая суммы. Поскольку запоминать все значения функции и промежуточные суммы нет необходимости, целесообразно для накопления суммы использовать формулу $z = z + y$, где знак "=" – знак присваивания, который означает, что вычисляемое значение выражения $z + y$ присваивается переменной z как новое ее значение. Если перед циклом задать начальное значение z равным нулю, то после первого выполнения цикла z будет равно первому слагаемому, после второго – сумме первого и второго и т. д. После окончания цикла z будет равно сумме всех вычисленных слагаемых.

Алгоритм решения задачи

1. Ввод значений массива x_i. 2. Задание начального значения суммы $z = 0$. 3. Организация цикла с параметром i, изменяющимся от 1 до 20 с шагом 1. 4. Накопление суммы $z = z + \frac{x_i}{i}$. 5. Конец цикла. 6. Вывод результата z.

Схема программы, описывающая алгоритм, представлена на рис. 9.

П. 10. Накопление произведения

Математическая формулировка задачи

Вычислить значение $z = \prod_{i=1}^{15} \frac{n+i}{i}$.

Аналогично сумме накапливается и произведение, с той лишь разницей, что для его накопления используется формула $z = zy$, а начальное значение произведения должно быть равно единице.

Алгоритм решения задачи

1. Ввод значения n. 2. Задание начального значения произведения $z = 1$. 3. Организация цикла с параметром i, изменяющимся от 1 до 15 с шагом 1. 4. Накопление произведения $z = z \frac{n+i}{i}$. 5. Конец цикла. 6. Вывод результата z.

Схема программы, описывающая алгоритм, представлена на рис. 10.

П. 11. Вычисление суммы бесконечного ряда

Математическая формулировка задачи

Вычислить сумму членов бесконечного ряда $z = 1 + x + \frac{x^2}{2!} + \ldots + \frac{x^n}{n!} + \ldots$ с точностью до члена ряда $\frac{x^n}{n!} \leqslant \epsilon$.

Вычисление суммы бесконечного ряда осуществляется с заданной точностью, которая определяется по члену ряда, меньшего некоторой величины ϵ. Заранее не-

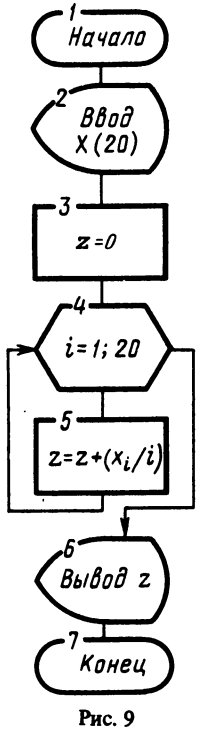


Рис. 9

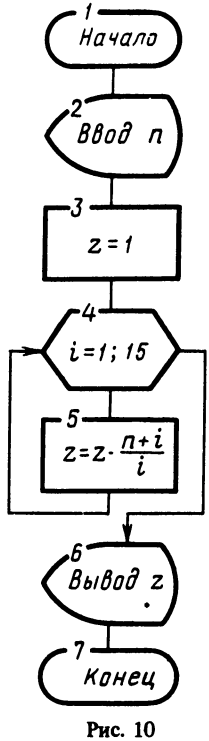


Рис. 10

известно, при каком члене ряда будет достигнута требуемая точность, поэтому это цикл с неизвестным числом повторений. Выход из цикла осуществляется по достижению требуемой точности. Для вычисления результата используется прием накопления суммы.

Вычислять текущий член ряда непосредственно по формуле $y = x^n/n!$ нецелесообразно, так как требуются большие затраты времени. Его удобно вычислять через предыдущий член, используя формулу $y_i = y_{i-1}x/i$. Следовательно, для вычисления любого члена ряда с номером i можно использовать формулу $y = yx/i$ (прием накопления произведения).

Алгоритм решения задачи

1. Ввод исходных данных x, ϵ. 2. Задание начального значения суммы $z = 1 + x$, так как вычислять два первых члена ряда нет необходимости. 3. Задание начального значения члена ряда $y = x$. 4. Задание начального значения параметра цикла $i = 2$. 5. Вычисление текущего члена ряда $y = yx/i$. 6. Накопление суммы $z = z + y$. 7. Изменение параметра цикла $i = i + 1$. 8. Проверка условия $y \leqslant \epsilon$; если оно не выполняется, то переход к началу цикла (вычислению текущего члена ряда), в противном случае выход из цикла. 9. Вывод результата z.

Схема программы, описывающая алгоритм, представлена на рис. 11.

П. 12. Вычисление многочлена.

Математическая формулировка задачи

Вычислить значение многочлена $y = 5x^7 - x^6 + 3x^4 - 2x + 1$.

Многочлен $y = a_1x^n + a_2x^{n-1} + \ldots + a_nx + a_{n+1}$ удобно вычислять по формуле Горнера $y = (\ldots(a_1x + a_2)x + \ldots + a_n)x + a_{n+1}$. Если выражение, стоящее в скобках, обозначить y_i, то выражение в следующих скобках можно вычислить по формуле $y_{i+1} = y_ix + a_{i+1}$. Значение многочлена получится после выполнения цикла n раз. Начальное значение y следует брать равным a_1 – коэффициенту при x в высшей степени, а цикл начинать при значении параметра $i = 2$. Все коэффициенты многочлена и свободный член сводятся в массив, состоящий из $n + 1$ элементов. Если многочлен не содержит членов с некоторыми степенями, то соответствующие коэффициенты равны нулю.

Коэффициенты многочлена сводятся в массив A (5, –1, 0, 3, 0, 0, –2, 1), количество элементов массива 8.

Алгоритм решения задачи

1. Ввод значений x и элементов массива A. 2. Задание начального значения многочлена $y = a_1$. 3. Организация цикла с параметром i, изменяющимся от 2 до 8 с шагом 1. 4. Вычисление $y = yx + a_i$. 5. Конец цикла. 6. Вывод результата y.

Схема программы, описывающая алгоритм, представлена на рис. 12.

П. 13. Нахождение наибольшего и наименьшего значений вычисляемой функции.

Математическая формулировка задачи

Найти наибольшее значение функции $y = xe^{bx-x^2}$ при изменении аргумента x от 0 до 4 с шагом 0,1.

Наибольшее и наименьшее значения функции вычисляются в том же цикле, что и текущее ее значение. Вычисленное значение функции сравнивается с наибольшим или наименьшим из всех предыдущих значений функции: если оно больше наибольшего или меньше наименьшего, то считается новым значением наибольшего или наименьшего соответственно. В противном случае значение наибольшего или наименьшего сохраняется. Нахождение наибольшего и наименьшего описывается следующими выражениями:

$$y_{max} = \begin{cases} y_i, \text{ если } y_i > y_{max}; \\ y_{max}, \text{ если } y_i \leqslant y_{max}. \end{cases} \quad (1)$$

$$y_{min} = \begin{cases} y_i, \text{ если } y_i < y_{min}; \\ y_{min}, \text{ если } y_i \geqslant y_{min}. \end{cases} \quad (2)$$

После первого выполнения цикла y_{max} и y_{min} должны быть равны y_1, тогда после второго выполнения цикла можно найти наибольшее или наименьшее из y_1 и y_2. При первом выполнении цикла y_{max} и y_{min} не определены, следовательно, перед циклом необходимо задать их начальные значения. При этом начальное значение y_{max} должно быть малым числом, заведомо меньшим y_1, а начальное значение y_{min}, наоборот, числом заведомо большим y_1, так как в противном случае после первого выполнения цикла y_{max} и y_{min} могут не стать равными y_1. В качестве начального значения для y_{max} выбирается число примерно -10^{10} и меньше, а для y_{min} – число $+10^{10}$ и больше.

Выберем в качестве начального значения $y_{max} = -10^{10}$.

Алгоритм решения задачи

1. Ввод значения b. 2. Задание начального значения $y_{max} = -10^{10}$. 3. Организация цикла по x от 0 до 4 с шагом 0,1. 4. Вычисление

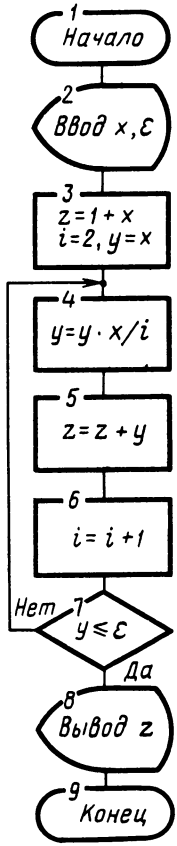


Рис. 11

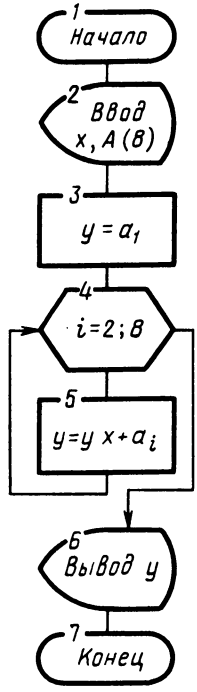


Рис. 12

$y = xe^{bx-x^2}$. 5. Проверка условия $y > y_{max}$, если оно выполняется, то $y_{max} = y$, в противном случае наибольшее значение остается прежним. 6. Конец цикла. 7. Вывод результата y_{max}.

Схема программы, описывающая алгоритм, представлена на рис. 13.

П. 14. Нахождение наименьшего значения в массиве

Математическая формулировка задачи

Найти наименьший элемент массива $x_1, x_2, ..., x_{40}$ и его порядковый номер.

При нахождении наибольшего или наименьшего среди элементов массива целесообразно в качестве начального значения наибольшего или наименьшего брать первый элемент массива. Поскольку сравнивать его с собой не имеет смысла, то цикл надо выполнять при изменении параметра, начиная со второго значения.

Поскольку здесь находится не только наименьший элемент, но и его порядковый номер, при выполнении условия $x_i < x_{min}$ необходимо не только присваивать $x_{min} = x_i$, но $i_{min} = i$, где i_{min} – номер наименьшего элемента. При задании начального значения $x_{min} = x_1$ необходимо задать начальное значение $i_{min} = 1$, так как может оказаться, что x_1 является наименьшим элементом массива.

Алгоритм решения задачи

1. Ввод значений элементов массива x. 2. Задание начальных значений x_{min} и i_{min}. 3. Организация цикла по i от 2 до 40 с шагом 1. 4. Проверка условия $x_i < x_{min}$; если оно выполняется, то вычисление $x_{min} = x_i$ и $i_{min} = i$, в противном случае переход к началу цикла. 5. Конец цикла. 6. Вывод результатов x_{min}, i_{min}.

Схема программы, описывающая алгоритм, представлена на рис. 14.

П. 15. Нахождение наименьшего значения функции, имеющей один минимум.

Математическая формулировка задачи.

Найти наименьшее значение функции $y = e^{ax+|c|x^2}$, имеющей один минимум при изменении аргумента x от 0 до 4 с шагом 0,2.

Рассмотренный алгоритм в п. 13 нахождения наибольшего и наименьшего значения универсальный, так как позволяет решить задачу, даже если функция не имеет максимума или минимума или имеет их несколько. Если же известно, что функция y имеет, например, один максимум, то время решения задачи можно существенно сократить, изменив алгоритм. В этом случае значения функции, лежащие до максимума, всегда будут больше, чем наибольшее из всех предыдущих (рис. 15), т. е. всегда будет выполняться условие $y_i > y_{max}$. После прохождения максимума функция начинает убывать, следовательно, всегда будет выполняться условие $y_i < y_{max}$. Причем, как только условие $y_i < y_{max}$, будет выполнено, необходимо прекращать вычисления, так как среди последующих значений функции не может быть наибольшего. Этот процесс можно описать условной формулой

$$y_{max} = \begin{cases} y_i, \text{ если } y_i > y_{max}, \\ \text{выход из цикла, если } y_i \leqslant y_{max}. \end{cases}$$

Аналогично, для нахождения наименьшего значения:

$$y_{min} = \begin{cases} y_i, \text{ если } y_i < y_{min}, \\ \text{выход из цикла, если } y_i \geqslant y_{min}. \end{cases}$$

Алгоритм решения задачи

1. Ввод значений a, c. 2. Задание начального значения наименьшего $y_{min} = 10^{10}$. 3. Организация цикла по x, изменяющегося от 0 до 4 с шагом 0,2. 4. Вычисление y. 5. Проверка условия $y < y_{min}$; если оно выполняется, то вычисление $y_{min} = y$, в противном случае выход из цикла. 6. Конец цикла. 7. Вывод результата y_{min}.

Схема программы, описывающая алгоритм, представлена на рис. 16.

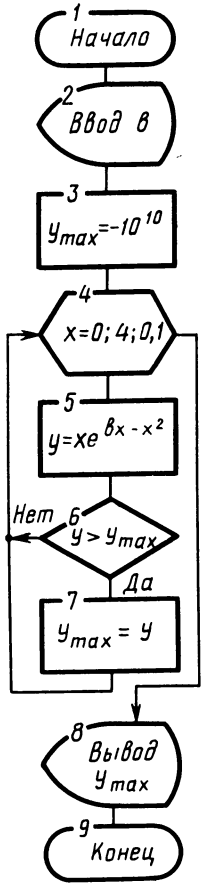


Рис. 13

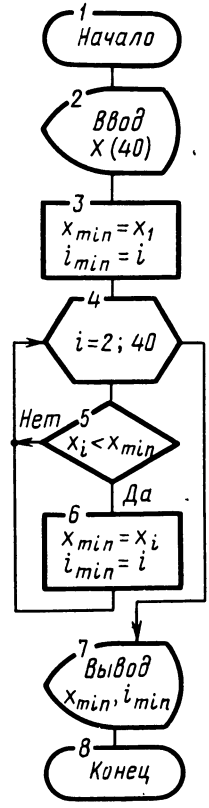


Рис. 14

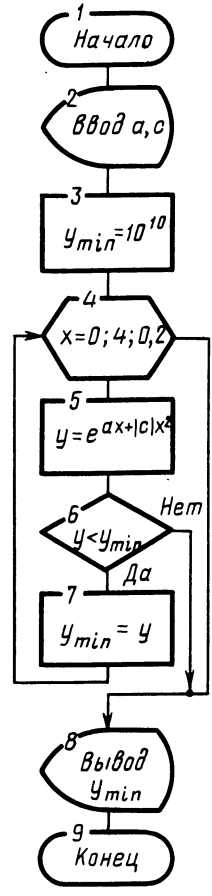


Рис. 16

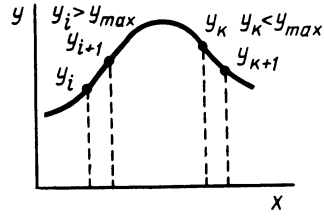


Рис. 15

ЗАДАЧИ ДЛЯ САМОСТОЯТЕЛЬНОГО СОСТАВЛЕНИЯ АЛГОРИТМОВ РЕШЕНИЯ

Алгоритмы линейной структуры

- **1.** Определить общую длину l трубы и число элементов n каждой секции в теплообменном аппарате типа "труба в трубе", если известны: поверхность F теплообмена, количество секций m, длина одного элемента l_1, а также наружный и внутренний диаметры $d_н$ и $d_в$.
- **2.** Рассчитать скорость движения V раствора в скважине с производительностью Q насоса и площадью поперечного сечения F скважины.
- **3.** Определить расчетную массу Q состава на подъеме с уклоном i (отношение высоты подъема к длине участка, умноженное на 1 000) и кривой радиуса R для тепловоза ТЭ3 с силой тяги F. Состав сформирован из четырехосных груженых вагонов массой q. Масса тепловоза P, удельное сопротивление тепловоза W_0, скорость состава V.
- **4.** Определить, за какое время t при электролизе медного купороса на электроде выделится масса меди m при прохождении тока I_1, если известно, что при прохождении тока $I_1 + I_2$ масса меди m выделяется на электроде через время t_1. Если половину этой массы $m/2$ получить при прохождении тока I_1, а вторую половину при прохождении тока I_2, то вся масса выделится за время t_2.
- **5.** Определить диаметры окружности выступов d_a и впадин d_f пары зубчатых колес, если известны передаточное число n, модуль m и межосевое расстояние a.
- **6.** Определить объем фильтрата V, прошедшего через 1 м^2 фильтра за время t, если известны коэффициент фильтрации c, характеризующий гидравлическое сопротивление фильтра, коэффициент фильтрации k, характеризующий режим фильтрации и физико-химические свойства осадка и жидкости.
- **7.** Найти зависимость массы C сухого осадка на 1 м^3 фильтрата от плотности фильтрата γ, концентрации суспензии x и влажности осадка m.
- **8.** Определить продолжительность фильтрации объема V_0 литров жидкости через 1 м^2 фильтра, если при испытаниях было собрано фильтрата V_1 через t_1 минут и V_2 через t_2 минут после начала фильтрации.
- **9.** Определить требуемую поверхность F барабанного вакуум-фильтра непрерывного действия и число его оборотов в минуту n, если на него подается суспензия (v_c), содержащая x процентов твердой фазы. Требуемая конечная влажность осадка $x_{вл}$. Предполагаемый вакуум на заводе P. Во время опытной фильтрации на лабораторной модели при вакууме P^* было установлено, что необходимая влажность осадка была достигнута за время τ работы зоны фильтрации. При этом константы фильтрации,

отнесенные к 1 м^2, оказались равными k^* и c. Плотность суспензии γ_c, плотность фильтрата $\gamma_ф$.

Алгоритмы разветвляющейся структуры

● **10.** Вычислить значение полного сопротивления W движению поезда, состоящего из основного сопротивления составу и сопротивления от уклона и кривизны пути, если известны: масса электровоза P; масса состава Q; количество вагонов n; скорость V поезда; радиус кривой R; длина кривой S; длина поезда L; величина уклона i – отношение высоты подъема к длине участка, умноженное на 1000.

● **11.** Определить, при каком максимальном значении крутизны пути i электровоз может взять с места состав с массой Q и количеством осей k.

Оси вагонов могут быть выполнены на подшипниках скольжения и роликовых подшипниках.

● **12.** Определить, при какой скорости движения вагона основное удельное сопротивление w_0 не будет превосходить заданное значение w_g.

● **13.** Определить режим течения жидкости в межтрубном пространстве теплообменника типа "труба в трубе" при следующих условиях: внутренняя труба теплообменника имеет наружный диаметр d_1, толщину стенки трубы h_1, наружная труба – диаметр d_2, толщину стенки h_2, массовый расход жидкости G, плотность γ, вязкость μ.

▲ ● **14.** Вычислить коэффициент α теплоотдачи для воды, подогреваемой в трубчатом теплообменнике, состоящем из труб с внутренним диаметром d. Вода по трубам идет со скоростью w и подогревается от температуры $t_{нач}$ до $t_{кон}$. Температура стенки $t_{ст}$, длина трубы L.

● **15.** Определить содержание углерода и тип материала по площади шлифа, занимаемого перлитом $F_п$ и цементитом $F_ц$ в процентах относительно полной площади шлифа.

Алгоритмы циклической структуры

● **16.** Определить зависимость глубины Y закаленного слоя (глубины проникновения тока) от частоты тока f.

● **17.** Для условия задачи 8 определить зависимость продолжительности фильтрации t от объема фильтрата V.

▲ ● **18.** Стенка печи состоит из двух слоев: огнеупорного кирпича толщиной h_1, строительного кирпича толщиной h_2. Температура внутри печи t_0, температура окружающего пространства t_3. Определить зависимость потерь теплоты q с 1 м^2 стенки печи и температуры t_2 на границе между огнеупорным и строительным кирпичом от изменения температуры внутри печи, а также значение t_0, при котором температура между слоями кирпича становится больше допустимой. Коэффициент теплоотдачи от печных газов к стенке α_1, а от стенки к воздуху α_2. Коэффициент теплопроводности огнеупорного кирпича λ_1, строительного кирпича λ_2.

● **19.** Определить скорость сушки в камерной сушилке в зависимости от времени и найти критическое влагосодержание материала. Анализ

проб на влажность производился в период времени от 0 до 20 ч с интервалом $\Delta t = 1$ ч. Влагосодержание материала в процентах от сухого вещества W_i сведено в массив W.

20. Найти наибольшее, наименьшее и среднее значения ударной вязкости стали по результатам эксперимента определения ударной вязкости в зависимости от температуры отпуска при медленном охлаждении. Результаты эксперимента сведены в массив X(50).

● **21.** Определить потребное количество глины x, необходимой для приготовления 1 м^3 раствора при изменении плотности раствора от 1 до 2,5 с шагом 0,1.

● **22.** Определить расчетную массу состава на подъеме с уклоном i (отношение высоты подъема к длине участка, умноженное на 1000) и кривой радиуса R для тепловоза ТЭ3. Состав сформирован из четырехосных груженых вагонов массой q; масса тепловоза $P = 252$ т.

Зависимость силы тяги F и удельного основного сопротивления W_0 от скорости V задается таблично.

● **23.** Определить расход A электрической энергии за время t на движение моторного вагона с учетом расхода энергии на собственные нужды.

● **24.** Вычислить среднее значение НВ измерений твердости по Бринеллю, используя формулу

$$\text{НВ} = \frac{2P}{\pi D(D - \sqrt{D^2 - d^2})},$$

где P – нагрузка; D – диаметр шарика; d – диаметр отпечатка.

Диаметры десяти отпечатков сведены в таблицу.

● **25.** Определить средние взвешенные значения горных пород по твердости $\bar{T}$ и абразивности $\bar{A}$.

● **26.** Определить среднее арифметическое значение X и среднеквадратическое отклонение S твердости металла. Значения твердости заданы таблицей.

● **27.** Вычислить координаты x_m, y_m, z_m центра тяжести системы материальных точек с массами $m_1, \ldots, m_n$ и координатами (x_1, y_1, z_1), $(x_2, y_2, z_2), \ldots, (x_n, y_n, z_n)$.

▲● **28.** Определить поверхность F теплообменника, в котором горячее поглотительное масло в количестве G охлаждается от T_1 до T_2 градусов за счет холодного масла, которое нагревается от t_1 до t_2 градусов (теплоемкость масла c). Коэффициент теплопередачи k в зависимости от температуры горячего масла T, изменяющейся от T_1 до T_2 с шагом $\Delta T = -10°\text{C}$, задан в таблице.

● **29.** Вычислить общее сопротивление цепи при параллельном включении n резисторов.

● **30.** Вычислить общее сопротивление цепи при параллельном включении n резисторов, используя формулу

$$R_0 = \frac{R_1 R_2 \ldots R_n}{R_2 R_3 \ldots R_n + R_1 R_3 \ldots R_n + \ldots + R_1 R_2 \ldots R_{n-1}} = \frac{\prod_{i=1}^{n} R_i}{\sum_{i=1}^{n} \frac{\prod_{i=1}^{n} R_i}{R_i}}.$$

● **31.** Вычислить, сколькими способами можно обеспечить снабжение N потребителей электрической энергией, если каждый из N источников может снабжать только одного потребителя, а каждый потребитель может снабжаться только от одного источника.

● **32.** Вычислить, сколькими способами можно обеспечить снабжение потребителей электроэнергией, если из n имеющихся источников могут быть использованы только $m (n > m)$.

▲● **33.** Определить зависимость КПД насоса как функцию производительности и его максимальное значение, если при испытании центробежного насоса с числом оборотов 1200 при перекачивании жидкости с плотностью γ были получены значения мощности N, потребляемой насосом, и создаваемого напора H в зависимости от производительности Q, сведенные в таблицы испытаний.

▲● **34.** Определить количество Q воздуха, необходимое для проветривания шахты (по газовыделению, по расходу взрывчатых веществ, по количеству людей), депрессию выработок h_2 и концентрацию метана h_1.

▲● **35.** Определить метацентрический радиус r, если известно объемное водоизмещение V судна и массив ординат y_i – расстояние от продольной оси до борта, ΔL – расстояние между шпангоутами.

▲● **36.** Определить мощность S, развиваемую цилиндром двигателя, по известной экспериментально-индикаторной диаграмме, дающей зависимость давления P от объема цилиндра V.

● **37.** Определить зависимость истинной скорости $V_и$ при заданной относительной скорости $V_о$ судна и скорости течения $V_т$ в зависимости от величины угла $q_т$ между диаметральной плоскостью судна и направлением течения. Повторить решение задачи при увеличении $V_о$ до $V_{о\ max}$ с шагом $\Delta V_о$.

▲● **38.** Определить зависимость просадки $\Delta T_к$ кормы судна от скорости V при заданных значениях осадки T судна, глубины судового хода H, длины L судна, ширины судна B.

▲● **39.** Определить, совпадают ли точки приложения равнодействующей сил веса G, заданной эпюрой весовой нагрузки (*1*), и равнодействующей силы поддержания P, заданной эпюрой сил поддержания воды (*2*) (рис. 17).

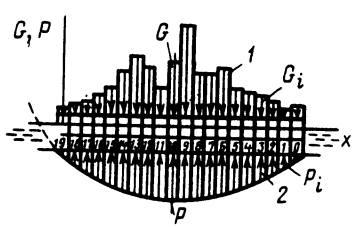


Рис. 17

ГЛАВА ВТОРАЯ

ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ БЕЙСИК

§ 3. ПРОСТЕЙШИЕ КОНСТРУКЦИИ ЯЗЫКА

▷ *В языке БЕЙСИК используются следующие символы:*

1. *Буквы латинского алфавита от A до Z.*
2. *Цифры от* 0 до 9.
3. *Специальные символы:* + – * / ↑ ␣ , . : ; ( ) =>< '
4. *Ключевые слова:*

LET – *пусть,* READ – *ввод,* DATA – *данные,* DIM – *размерность,* PRINT – *вывод,* STOP – *останов,* END – *конец,* TO – *к,* STEP – *шаг,* GOTO – *перейти,* NEXT – *конец цикла,* INPUT – *ввод с экрана,* DEFFN – *определение функции,* REM – *комментарий,* FOR – *для,* IF – *если,* GOSUB – *переход к подпрограмме,* RETURN – *возврат.*

5. *Знаки операций:*

а) *арифметических:* + (*сложение*), * (*умножение*), / (*деление*), – (*вычитание*), ↑ *или* ^ (*возведение в степень*);

б) *отношения:* > (*больше*), < (*меньше*), = (*равно*), >< *или* <> (*не равно*), >= (*больше или равно*), <= (*меньше или равно*).

К простейшим конструкциям языка относятся константы (числа), переменные, стандартные функции и выражения.

Константа представляет собой число со знаком "+" или "–". Положительный знак числа может опускаться в записи. Различают константы целого и действительного типов. Однако в БЕЙСИКе отсутствует четкое различие между числами целого и действительного типа. Для отделения целой части от дробной используется точка. Помимо естественной формы записи может использоваться запись чисел с десятичным множителем, порядок которого задается после буквы E целым числом.

Примеры записи:

Числа	*Запись на БЕЙСИКе*
2	2
0,65	0.65 или . 65
–11,426	–11.426
$2{,}6 \cdot 10^4$	2.6 E 4
0,00081	0.81E–3 или .81E–3

Переменная, представляющая символическое изображение величины, используется для записи выражений.

Различают простые переменные и переменные с индексом. Простые переменные записываются своими именами, которые образуются из буквы или буквы и цифры. В одной программе можно использовать 26 имен, образованных буквой, и 26x10 имен, полученных от комбинаций букв и цифр.

Примеры записи имен простых переменных:

X, X1, T, T0, T9, A4, I, K5

Переменные с индексами являются элементами массивов. Массив представляет собой упорядоченную запись значений, обозначенных одним именем. Расположение значения в массиве определяется индексами, которые записываются после имени в круглых скобках. Имя массива записывается одной буквой и в пределах одной программы можно использовать 26 массивов.

Различают одномерные массивы (векторы) и двухмерные (матрицы). Для размещения массива в памяти ЭВМ должно быть выделено соответствующее количество полей (ячеек) памяти. Одномерный массив (вектор) с именем D может быть представлен следующим образом:

Индекс элемента массива D	1	2	3	4	5
Значение элемента массива	0.25	1.1	1.3	1.65	2.1

Для размещения этого массива необходимо выделить пять ячеек памяти. Необходимо учитывать, что, как правило, начальным индексом является ноль. Использование нулевого индекса не является обязательным и в данном сборнике задач использоваться он не будет. Тогда первый, третий и пятый элементы соответственно будут записаны как D(1), D(3) и D(5).

При работе с изменяющимся индексом индекс указывается буквой и, следовательно, элемент массива следует записать как D(I).

В двухмерных массивах (матрицах) расположение элемента в таблице определяют два индекса: первый индекс указывает номер строки, а второй – номер столбца, на пересечении которых он располагается. Поэтому элемент массива C, находящийся на пересечении второй строки и третьего столбца, будет записан как C(2, 3). Элемент массива, у которого изменяются текущие значения индексов строк и столбцов, можно записать как C(I, J). Эта запись соответствует элементу массива, находящемуся на пересечении *I* строки и *J* столбца.

Наиболее часто используемые функции в задачах вычислительного характера определены в виде стандартных.

Обращение к **стандартным функциям** заключается в записи имени функции и аргумента, записанного в скобках. Аргументы тригонометрических функций должны задаваться в радианах. Перевод в радианы выполняется по формуле РАДИАНЫ=ГРАДУСЫ $\cdot \pi/180^\circ$.

▷ *В ЭВМ типа ДВК-2М, Д3-28, "Искра-226" используются следующие общие стандартные функции:* SIN(X) – $\sin x$, COS(X) – $\cos x$, TAN (X) – $\operatorname{tg} x$, ATN (X) – $\operatorname{arctg} x$, EXP(X) – e^x, LOG(X) – $\ln x$, ABS (X) – $|x|$, SQR (X) – $\sqrt{x}$, INT(X) – ближайшее целое к X; SGN(X) – знак X: +1, если $X > 0$; 0, если $X = 0$; –1, если $X < 0$.

▷ *Помимо перечисленных стандартных функций в каждой ЭВМ имеются дополнительные функции:* в ДВК-2М: RND(X) – *выбор случайного числа;* в Д3- 28: RND(X), DEG(X) – $(180X)/\pi$ – *перевод в градусы;* RAD(X) – $(\pi X)/180$ – *перевод в радианы;* ASN(X) – $\arcsin x$, ACS(X) – $\arccos x$, HSN – $\operatorname{sh} x$, HCS(X) – $\operatorname{ch} x$, HTN(X) – $\operatorname{th} x$, AHS(X) – $\operatorname{arsh} x$, AHC(X) – $\operatorname{arch} x$, AHT(X) – $\operatorname{arth} x$, EXT(X) – 10^X;

в "Искре-226": ARCSIN(X) – $\arcsin x$, ARCCOS (X) – $\arccos x$, POUND (x, n) – *выполнение округлений с заданной точностью, где* x – *округляемое значение;* n – *точность округления. Стандартная функция* RND *может не содержать аргумента и работает как генератор случайных чисел в диапазоне от 0 до 1. Использование этой функции в выражении эквивалентно замене этой функции полученным случайным числом.*

Выражение представляет собой компактную запись, состоящую из чисел, переменных, стандартных функций, разделенных скобками и знаками операций. Оно определяет действия и порядок их выполнения при вычислении искомого значения.

Примеры записи:

Математическая	*На БЕЙСИКе*
$ax^2 + b$	A * X ↑ 2 + B или A * X * X + B
$c - \sqrt{t^3 + 1}$	C – SQR(T ↑ 3 + 1)
$\dfrac{a \cdot \sin x + b}{c + d}$	(A * SIN(X) + B)/(C + D)

В отличие от математической записи выражений в языке БЕЙСИК следует пользоваться **следующими правилами.**

▷ 1. *Нельзя опускать знак операции умножения. Математическая запись* ab *должна быть оформлена выражением* A * B, *иначе образуется имя переменной* AB, *которое недопустимо синтаксисом языка.*

2. *Выражение в виде дроби записывается в одну строку с использованием знака операции деления. Запись* $(a + x)/b$ *следует записать выражением* (A + X)/B.

3. *Не допускается запись двух непосредственно следующих друг за другом знаков арифметических операций. Запись* $\dfrac{c}{-d}$ *должна быть оформлена выражением* C/(–D).

4. *Порядок выполнения арифметического выражения определяется скобками. В их отсутствии операции выполняются согласно старшинству (приоритету) операций в следующем порядке: вычисление значения функции; возведение в степень; умножение или деление; сложение или вычитание.*

5. *Все операции в выражении выполняются слева направо, за исключением операции возведения в степень, которая выполняется справа налево. Следовательно, математическая запись* x^{y^z}, *оформленная выражением* X ↑ Y ↑ Z, *будет выполнена эквивалентным выражением* X ↑ (Y ↑ Z).

6. *Операция возведения в целую степень выполняется через операцию многократного умножения, а в действительную – через логарифмическую функцию.*

§ 4. ОПЕРАТОРЫ ЯЗЫКА

Оператор является основной конструкцией языка. Различают выполняемые операторы, определяющие действия и порядок их выполнения. К ним относятся оператор присваивания, операторы ввода и вывода данных, операторы безусловного перехода, условный и цикла. Невыполняемые операторы описывают размещение, формирование и восстановление данных, оформление комментариев.

Запись оператора начинается с метки (номера строки), которая назначается самим пользователем.

Оператор присваивания служит для вычисления значения выражения и присваивания этого значения одной или нескольким переменным. Общий вид оператора

$$n \sqcup \text{LET} \sqcup V_1 = V_2 = \ldots = V_n = a,$$

где n **– метка (номер) оператора; LET – ключевое слово;** $V_1, V_2, \ldots, V_n$ **– имена переменных;** a **– арифметическое выражение.**

В некоторых версиях языка оператор присваивания записывается без LET. В данном задачнике это ключевое слово не используется.

К моменту обращения к оператору присваивания значения переменных, входящих в выражения, должны быть определены.

Примеры записи операторов:

```
10 P=3.14159
20 X=0.75
30 I=1
40 C=A*EXP(Y)
```

С помощью первых трех операторов присваивания переменные P, X и I получат соответственно значения 3.1415, 0.75 и 1. В операторе с меткой 50 переменная C получит значение выражения A * EXP(Y). В случае, если переменным S, T и W нужно присвоить нулевые значения, то можно записать три оператора

```
40 S=0
50 t=0
60 W=0
```

или записать в одном `70 S=T=W=0`

Особенности оператора присваивания ярко проявляются в операторе I = I + 1, в котором новое значение I (результат) получается из предыдущего добавлением единицы. Если задать начальное значение I = 0 и выполнить несколько раз оператор присваивания I = I + 1, то можно получить счетчик единиц. Следовательно, **с помощью оператора присваивания можно:**

▷ а) *задавать значения переменным*

```
10 X=0
20 I=0
```

б) *наращивать значения переменных*

```
30 X=X+0.1
40 I=I+2
```

в) *избавляться от переменной с индексом, так как индексная переменная обрабатывается ЭВМ больше по времени, чем простая переменная, и записывается длиннее.*

Например, в математической зависимости

$$W = \frac{a_i e^{a_i} - \sin a_i}{1 + \sqrt{a_i}}$$

переменная с индексом используется несколько раз. Можно эту зависимость записать одним оператором присваивания:

```
20 W=(A(I)*EXP(A(I))-SIN(A(I)))/(1+SQR(A(I)))
```

или с помощью двух операторов:

```
20 C=A(I)
30 W=(C+EXP(C)-SIN(C))/(1+SQR(C))
```

(вторая запись является предпочтительней);

г) *исключать повторяющиеся участки вычислений, встречающиеся в выражениях.*

Например, в математической зависимости

$$y = \frac{x^2 + 1{,}4\,x^2 e^{x^2}}{e^{x^2} - \sin x^2}$$

пять раз встречается вычисление x^2 и два раза e^{x^2}.

В этом случае следует записать три оператора присваивания

```
100 X2=X*X
110 E=EXP(X2)
120 Y=(X2+1.4*X2*E)/(E-SIN(X2))
```

Время вычислений будет меньше за счет однократного вычисления повторяющихся вычислений.

В БЕЙСИКе имеется ряд операторов для ввода данных. Общий вид совместно используемых операторов

n_1 DATA $C_1, C_2, C_3 \ldots$

n_2 READ $V_1, V_2, V_3, \ldots$

где n_1, n_2 – метки операторов; C_1, C_2, C_3, ... – числовые значения переменных, указанных в операторе READ; V_1, V_2, V_3 – имена переменных, значения которых подлежат вводу в ЭВМ.

Оператор DATA (данные) формирует числовой массив исходных данных. Оператор является невыполняемым и может записываться в любом месте программы до последнего оператора END.

В операторе READ (ввод) указываются имена переменных. Между

именами переменных в операторе READ и числовыми значениями в операторе DATA должно быть взаимооднозначное соответствие.

Например, после выполнения операторов

```
10 DATA 0.8,1.2,3.6
20 READ A,B,C
```

переменные получат значения $A = 0{,}8$, $B = 1{,}2$, $C = 3{,}6$.

В программе может быть несколько операторов DATA, числовые значения которых образуют блок данных. Первыми в блок записываются числовые значения оператора DATA, имеющего наименьшую метку (номер строки).

Например, операторы

```
20 DATA 1,2,6
.................
80 DATA 3,8
.................
120 DATA 5,4,7
```

образуют числовой блок данных 1, 2, 6, 3, 8, 5, 4, 7.

Этот же блок данных может быть сформирован одним оператором

```
20 DATA 1,2,6,3,8,5,4,7
```

Из сформированного блока данных выборка числовых значений может осуществляться одним или несколькими операторами READ.

Например, выборка производится одним оператором:

```
40 READ A,B,C,D,E,F,P,T
```

или выборка осуществляется двумя операторами:

```
60 READ A,B,C
100 READ D,E,F,P,T
```

В обоих случаях переменные получат следующие значения: A = 1, B = 2, C = 6, D = 3, E = 8, F = 5, P = 4, T = 7.

Оператор DATA должен записываться один на строке.

Оператор RESTORE (восстановить) служит для восстановления значений в блоке данных, образованном операторами DATA.

Общий вид записи *n* RESTORE, где *n* – метка оператора.

После выполнения оператора RESTORE операторами READ может осуществляться повторная выборка значений из блока данных. Оператор RESTORE может записываться в программе в любом месте, не дожидаясь полной выборки из блока данных.

Например,

```
30 DATA 0.6,1.2,2.8
40 READ X,Y,Z
.................
120 RESTORE
130 READ Y,Y,Z
```

Оператором DATA будет сформирован блок данных, содержащий значения 0.6, 1.2 и 2.8. После выборки этих значений оператором READ блок данных освобождается. Для повторного использования значений, записанных в операторе DATA, необходимо их восстановить с помощью оператора RESTORE.

В дальнейшем в программе требуются значения только переменных Y и Z. Однако блок данных допускает только последовательную выборку данных начиная с первого значения. Поэтому для установления соответствия между операторами DATA и READ необходимо использовать в операторе READ имена трех переменных. Оператором READ с меткой 130 переменной Y вначале будет присвоено фиктивное значение 0.6, а затем действительное значение 1.2.

Задание значений переменным Y и Z из блока данных можно было бы организовать, введя в оператор READ фиктивную переменную R.

Например,

```
130 READ R,Y,Z
```

Переменная R в программе может не использоваться, но ее введение позволяет установить соответствие между значениями, записанными в блоке данных, и именами переменных, указанных в операторе READ.

Оператор INPUT служит для ввода данных с экрана дисплея.

Общий вид оператора *n* INPUT V_1, V_2, V_3, ..., где *n* – метка оператора; V_1, V_2, V_3, . . . – имена переменных, значения которых вводятся с экрана дисплея.

При выполнении оператора INPUT машина приостанавливает выполнение программы и предоставляет пользователю возможность задать значения переменным. Между именами переменных и их значениями должно быть взаимооднозначное соответствие по количеству и месту расположения. Оператор INPUT может записываться в любом месте программы.

Оператор PRINT (вывод) служит для вывода результатов вычислений, результатов вычислений со своими именами, пояснений или заголовков к результатам.

Общий вид записи оператора *n* PRINT < список >, где *n* – метка оператора, < список > – константы, имена переменных, выражения, текст, функция ТАБ (X).

Элементы списка могут отделяться друг от друга запятой или точкой с запятой. При использовании оператора в виде *n* ␣ PRINT ␣ V_1, V_2, V_3 строка разделяется на пять зон по 14 позиций в каждой из них. Количество зон и число позиций в зоне зависит от типа ЭВМ. Первые позиции каждой зоны (1, 15, 29, 43, 57) отводятся для вывода знаков отрицательных чисел, у положительных чисел эти позиции остаются свободными. Например, при выводе переменных со значениями A = 2.6, B = −11.25, C = = 8.4 оператор вывода и строка результатов будут иметь вид

```
70 PRINT A,B,C
   2.6     -11.25      8.4
```

Если в списке выводимых величин больше пяти, то автоматически осуществляется переход на новую строку. Например, оператор PRINT 1, 2, 3, 4, 5, 6, 7 позволит получить размещение констант в следующем виде:

1-я поз.	15-я поз.	29-я поз.	43-я поз.	57-я поз.
␣ 1	␣ 2	␣ 3	␣ 4	␣ 5
␣ 6	␣ 7			

Помимо имен переменных в списке оператора могут записываться выражения. В этом случае вначале вычисляется значение выражения и затем выводится результат. Например,

```
60 M=24
70 N=3
80 PRINT M,N,M/N
```

После вывода в первых трех зонах будут размещены значения

24 3 8

Для компактного вывода разделителем элементов списка является символ точка с запятой (;). При этом каждое последующее значение выводится через два пробела после предыдущего.

Например, оператор

```
100 PRINT 5;4;3;2;1;2;3;4;5;
```

вызовет размещение чисел в следующем виде:

5 ␣ ␣ 4 ␣ ␣ 3 ␣ ␣ 2 ␣ ␣ 1 ␣ ␣ 2 ␣ ␣ 3 ␣ ␣ 4 ␣ ␣ 5.

В случае, если элементы списка не помещаются в одной строке, то элементы списка можно распределить на два оператора.

Например, операторы

```
60 PRINT C,D,A↑2+SQR(C+D),
70 PRINT A
```

эквивалентны действию одного оператора

```
100 PRINT C,D,A↑2+SQR(C+D),A
```

Наличие или отсутствие разделителей элементов списков (запятая, точка с запятой) в конце оператора PRINT позволяет по-разному размещать результаты при выводе.

Если после элемента списка ставится запятая, то следующее значение выводится в той же строке.

Например, операторы

```
40 PRINT 8,12 ,
60 PRINT 7,2,3
```

позволяют получить размещение значений в одной строке

8 12 7 2 3

Наличие дополнительной запятой между элементами списка вызывает пропуск одной зоны.

Например,

```
10 DATA 6,,12,7
20 READ A,B,C
30 PRINT A,B,C
```

оператор вывода вызовет пропуск одной зоны между переменными A и B

6 12 7

Использование оператора PRINT без элементов списка позволяет получить пропуск строки.

```
10 DATA 1,2,3,4,5
20 READ A,B,C,D,T
30 PRINT A,B,C
40 PRINT
50 PRINT D,T
```

Операторы вывода позволят получить значения в виде трех строк

1-я строка 1 2 3

2-я строка

3-я строка 4 5

В операторе PRINT среди элементов списка можно записывать комментарии к результатам вычислений или имена выводимых значений.

Например, операторы

```
70 PRINT 'КОРНИ КВАДРАТНОГО УРАВНЕНИЯ'
80 PRINT 'X1=';X1,'X2=';X2
```

позволят получить следующее размещение выводимой информации при значениях X1 = 0.8, X2 = 1.2.

КОРНИ КВАДРАТНОГО УРАВНЕНИЯ

XI = ␣␣ 0.8 ␣␣␣␣␣␣ X2 = ␣␣ 1.2

Оператор REM (REMARK) используется для записи комментариев (пояснений) в программах пользователей. В комментарии могут содержаться любые символы языка, в том числе и буквы русского алфавита.

Например, программа для вычисления корней уравнения может содержать комментарий:

```
10 REM ВЫЧИСЛЕНИЕ КОРНЕЙ УРАВНЕНИЯ
```

Оператор является невыполняемым и не оказывает действий в выполняемой программе. Однако нужно помнить, что комментарии занимают поля памяти. При записи оператора REM в одной строке с другими операторами он должен стоять последним, так как все, что записано в строке после оператора REM, игнорируется.

Операторы STOP и END служат для прекращения вычислений по программе.

Операторов STOP может быть несколько в программе и они могут быть размещены в разных точках программы. Чаще всего оператор STOP используется при отладке программы.

Оператор END используется в программах многократного использования. Он должен быть последним в программе (иметь наибольшее значение метки).

Общий вид записи операторов n_1 STOP и n_2 END.

Естественный порядок выполнения операторов определяется номерами строк (метками).

В зависимости от условий задачи порядок вычисления может быть изменен с помощью операторов безусловного перехода GOTO и условного оператора IF.

Оператор безусловного перехода имеет общий вид записи n GOTO n_1, где n_1 – номер строки, к которой осуществляется переход, например 60 GOTO 180. Это значит, что оператор с номером строки 60 передает управление оператору с номером строки 180.

Условный оператор имеет два вида записи:

n IF $a \odot b$ THEN n_1,

n IF $a \odot b$ THEN S, где a, b – арифметические выражения; $\odot$ – знак операции отношения: =, <, <=, >, >=, <>; n_1 – номер строки (метка) оператора, к которому осуществляется переход; S – любой один оператор языка.

Работа оператора заключается в следующем: вычисляются арифметические выражения и проверяется условие. Если условие выполняется, то осуществляется переход к оператору с указанным номером строки или выполняется записанный оператор. При невыполнении условия выполняется оператор, стоящий за условным.

Например,

```
80 IF 2.5*X-C>B*X THEN 160
90 PRINT 'УСЛОВИЕ НЕ ВЫПОЛНЕНО'
100 ...
```

В случае, если 2.5 * X – C больше B * X, то осуществляется переход к оператору с меткой 160, а при невыполнении условия будет выведено сообщение – УСЛОВИЕ НЕ ВЫПОЛНЕНО.

В примере

```
60 IF Y<2.6 THEN A=1.5
70 B=1.8
```

при выполнении условия (Y < 2.6) переменным A и B будут соответственно присвоены значения 1.5 и 1.8. Если условие не будет выполнено, то только B получит значение 1.8.

Оператор DIM служит для резервирования места в памяти ЭВМ под массивы. Индексы одномерных массивов начинаются с (0), а двухмерных – с (0,0).

Например, для описания одномерного массива D из 15 элементов и матрицы C, расположенной в четырех строках и пяти столбцах с учетом нулевых индексов элементов, оператор DIM будет иметь вид

```
10 DIM D(14),C(3,4)
```

Так как использование нулевых индексов не является обязательным, то в данном сборнике задач они использоваться в основном не будут, тогда оператор DIM для предыдущего условия будет записан

```
10 DIM D(15),C(4,5)
```

Наибольшее значение индекса элемента массива – 255,

Операторы FOR и NEXT обеспечивают многократное выполнение операторов, заключенных между ними (организуют цикл).

Общий вид записи операторов

n_1 FOR $V = E_1$ TO E_2 STEP E_3

< операторы цикла >

n_k NEXT V,

где V – имя управляющей переменной целого или действительного типа; E_1, E_2, E_3 – выражения, определяющие соответственно начальное значение, конечное значение и шаг изменения переменной V.

Если шаг изменения переменной (E_3) равен +1, то его можно не указывать:

n FOR $V = E_1$ TO E_2.

Например, фрагмент программы

```
10 A=1
20 C=2
30 FOR B=1 TO 6
40 T=(A+B)/C
50 PRINT T
60 NEXT B
```

позволит получить результаты

1
1.5
2
2.5
3
3.5

Допустимы положительные и отрицательные значения шага. Например, два фрагмента программ позволят получить следующие результаты:

```
50 FOR C=2 TO 8 STEP 2
60 PRINT C;
70 NEXT C
```

```
50 FOR C=8 TO 2 STEP -2
60 PRINT C;
70 NEXT C
```

Вычисления в цикле продолжаются до тех пор, пока значение управляющей переменной при положительном шаге не превысит конечное значение, а при отрицательном – не станет меньше конечного значения. После этого управление передается оператору со следующим значением метки после оператора NEXT.

Подпрограмма, записываемая пользователем, представляет собой группу операторов, к которой можно обращаться многократно из разных точек программы.

Обращение к первому оператору подпрограммы осуществляется оператором обращения к подпрограмме GOSUB.

Общий вид записи: n GOSUB n_1, где n – метка оператора в программе; n_1 – метка первого оператора подпрограммы.

Подпрограмма должна заканчиваться оператором выхода из подпрограммы (RENURN).

Подпрограммой удобно воспользоваться в тех случаях, когда встречаются повторяющиеся участки вычисления в программе, состоящие из

нескольких операторов. При организации программы с подпрограммой необходимо: 1) перед обращением к подпрограмме задать с помощью оператора присваивания фактические значения формальным параметрам, использующимся в подпрограмме; 2) после выхода из подпрограммы запомнить результаты вычисления путем присваивания вычисленного значения фактическому параметру.

Помимо стандартных функций пользователь может часто встречающуюся функцию в программе описать с помощью **оператора DEF**, а затем к ней многократно обращаться.

Общий вид оператора:

$$\text{DEF FN}\, a(q_1, q_2, \ldots, q_n) = b,$$

где a – имя функции; $q_1, q_2, \ldots, q_n$ – формальные параметры; b – выражение.

Оператор описания функции может располагаться в любом месте программы. Если этот оператор встречается до обращения к нему, то он пропускается.

Общие сведения о записи программ

Программа на языке БЕЙСИК состоит из строк, в которых записываются операторы или команды. Программа может выполняться в режимах: косвенном или непосредственном. При работе в косвенном режиме запись каждой строки начинается с метки (номера строки). Номера строк назначаются самим пользователем и могут принимать значения от 1 до 8191.

Номер строки выполняет три функции: 1) определяет порядок выполнения операторов; 2) служит для ссылки на данный оператор; 3) используется в сообщениях об ошибках.

Номера строк целесообразно записывать кратными 5 или 10, что обеспечит возможность вставлять при отладке забытые или пропущенные операторы или команды.

Общий вид записи оператора:

< МЕТКА> ␣ <ОПЕРАТОР> ВК (управляющий символ – возврат каретки)

При работе в непосредственном режиме операторы не содержат номеров строк и выполняются по мере ввода в ЭВМ.

§ 5. ПРИМЕРЫ СОСТАВЛЕНИЯ ПРОГРАММ

Программирование алгоритмов линейной структуры

П. 16. Составить программу для условия задачи П. 1. Схема программы решения приведена на рис. 1.

Каждый блок схемы программы описывается одним оператором, за исключением блока *2*, который необходимо описать операторами READ и DATA. Программа имеет следующий вид:

```
10 REM ПЛОЩАДЬ ТРЕУГОЛЬНИКА
20 DATA .....
30 READ A,B,C
40 P=(A+B+C)/2
50 S=SQR(P*(P-A)*(P-B)*(P-C))
60 PRINT S
70 END
```

Программирование алгоритмов разветвляющейся структуры

П. 17. Составить программу для условия задачи П. 3.

В программе блок *3* будет записан условным оператором. При этом нужно сохранить ту последовательность операторов, чтобы они соответствовали последовательности блоков на схеме рис. 2. Переход к оператору END после вывода на экран значения Z осуществляется оператором безусловного перехода GOTO. Программа решения задачи выглядит следующим образом:

```
05 REM ФУНКЦИЯ
10 DATA...
20 READ X,Y
30 IF X*Y=0 THEN 70
40 Z=1/X/Y
50 PRINT Z
60 GOTO 80
70 PRINT 'XY=0'
80 END
```

Программирование алгоритмов циклической структуры

П. 18. Составить программу для условия задачи П. 4.

Каждый блок программы записывается одним оператором. Программа, записанная с условным оператором, – вариант 1, а с оператором цикла – вариант 2.

Вариант 1

```
05 REM ЦИКЛ IF
10 X=0.1
20 Z=SIN(X)/X
30 PRINT Z
40 X=X+0.1
50 IF X<=1.01 THEN 20
60 END
```

Вариант 2

```
10 REM ЦИКЛ FOR
20 FOR X=0.1 TO 1 STEP 0.1
30 Z=SIN(X)/X
40 PRINT Z
50 NEXT X
60 END
```

П. 19. Составить программу для условия задачи П. 5.

Число повторений вычислений в цикле неизвестно, а следовательно, цикл нужно организовать с условным оператором, как показано на рис. 5.

Программа выглядит следующим образом:

```
05 REM ЧИСЛО ПОВТОРЕНИЙ
10 DATA ...
20 READ X,E
30 K=1
40 IF X↑K/K<=E THEN 70
50 K=K+1
60 GOTO 40
70 PRINT 'K=';K
80 END
```

П. 20. Организация цикла с несколькими одновременно изменяющимися параметрами. Составить программу для условия задачи П. 7.

Цикл с известным числом повторений удобнее организовать с использованием оператора цикла. Каждый блок схемы в программе записывается оператором. Для размещения массива B необходимо выделить 11 ячеек памяти, в которые будут введены значения с помощью оператора READ. Программа может быть записана в двух вариантах.

Вариант 1

```
10 DIM B(11)
20 FOR I=1 TO 11
30 READ B(I)
40 NEXT I
50 DATA...
60 A=1
70 FOR I=1 TO 11
80 Z=A*B(I)
90 PRINT Z
100 A=A+0.1
110 NEXT I
120 END
```

Вариант 2

```
05 REM ЦИКЛ
10 DIM B(11)
20 DATA...
30 A=1
40 FOR I=1 TO 11
50 READ B(I)
60 Z=A*B(I)
70 PRINT Z
80 A=A+0.1
90 NEXT I
100 END
```

Отличие вариантов заключается в организации ввода массива B. В первом варианте операторами 20, 30 и 40-м организован ввод всех значений массива, а с 60-го оператора выполняются вычисления. Во втором варианте ввод элементов массива B осуществляется в цикле, в котором выполняются вычисления. Программа получается короче.

П. 21. Запоминание результатов вычислений. Составить программу для условия задачи П. 8, выделив необходимое количество полей памяти под результирующий массив Z.

Для запоминания необходимо предусмотреть выделение нужного количества ячеек памяти с помощью оператора DIM. Чтобы вычисляемые элементы занимали соответствующие ячейки памяти, в операторе присваивания следует записывать переменную с индексом (элемент массива).

Программа имеет следующий вид:

Вариант 1

```
05 REM ЗАПОМИНАНИЕ
10 DIM X(10),Z(10)
20 FOR I=1 TO 10
30 READ X(I)
40 NEXT I
50 FOR I=1 TO 10
60 Z(I)=SIN(I*X(I))/I
70 NEXT I
80 FOR I=1 TO 10
90 PRINT Z(I)
100 NEXT I
110 END
```

Вариант 2

```
10 DIM X(10),Z(10)
20 FOR I=1 TO 10
30 READ X(I)
40 Z(I)=SIN(I*X(I))/I
50 NEXT I
60 FOR I=1 TO 10
70 PRINT Z(I)
80 NEXT I
90 END
```

П. 22. Накопление суммы. Составить программу для условия задачи П. 9.

Ввод массива X(20) будет осуществляться в основном цикле. В остальном операторы программы будут соответствовать блокам схемы (см. рис. 9).

В соответствии с алгоритмом вычисления суммы необходимо: задать начальное значение имени результата, равное нулю, с помощью оператора присваивания (Z =

= 0); организовать многократное повторение вычислений с помощью оператора цикла; в цикле с помощью оператора присваивания накапливать сумму по формуле $Z = Z + y_i$.

Программа будет выглядеть следующим образом:

```
05 REM СУММА
10 DIM X(20)
20 DATA ...
30 Z=0
40 FOR I=1 TO 20
50 READ X(I)
60 Z=Z+X(I)/I
70 NEXT I
80 PRINT 'Z=';Z
90 END
```

П. 23. Накопление произведения. Составить программу для условия задачи П. 10.

Все операторы программы соответствуют блокам схемы программы (см. рис. 10). В соответствии с алгоритмом накапливания произведения начальное значение имени результата равно единице (Z = 1), а в цикле многократно накапливается произведение оператором присваивания Z = Z * Y(I).

Исходные данные: N. Результат: Z.

Программа выглядит следующим образом:

```
05 REM ПРОИЗВЕДЕНИЕ
10 READ N
20 DATA ...
30 Z=1
40 FOR I=1 TO 15
50 Z=Z*(N+I)/I
60 NEXT I
70 PRINT 'Z=';Z
80 END
```

П. 24. Вычисление суммы бесконечного ряда. Составить программу для условия задачи П. 11.

Программа вычисления соответствует схеме алгоритма (см. рис. 11), блоку *3* на схеме будут соответствовать три оператора присваивания. Число повторений неизвестно, поэтому цикл будет организован с использованием операторов условного и безусловного перехода, либо, изменив условие в блоке *7* ($y > \epsilon$), только с одним условным оператором. Исходные данные: X, E (вместо ϵ); промежуточные переменные: y, I. Результат: Z.

Программа будет выглядеть следующим образом:

Вариант 1

```
05 REM РЯД
10 DATA ...
20 READ X,E
30 Z=1+X
40 I=2
50 Y=X
60 Y=Y*X/I
70 Z=Z+Y
80 I=I+1
90 IF Y<=E THEN 110
100 GOTO 60
110 PRINT 'Z=';Z
120 END
```

Вариант 2

```
05 REM РЯД
10 DATA ...
20 READ X,E
30 Z=1+X
40 I=2
50 Y=X
60 Y=Y*X/I
70 Z=Z+Y
80 I=I+1
90 IF Y>E THEN 60
100 PRINT'Z=';Z
110 END
```

П. 25. Вычисление многочлена. Составить программу для условия задачи П. 12.

Необходимо выделить требуемое количество полей памяти оператором DIM, ввод массива А организовать отдельно от цикла, в котором вычисляется результат (см. рис. 12).

Исходные данные: X, массив А. Результат: У.

Программа будет выглядеть следующим образом:

```
05 REM МНОГОЧЛЕН
10 DIM A(8)
20 DATA ...
30 READ X
40 DATA 5,-1,0,3,0,0,-2,1
50 FOR I=1 TO 8
60 READ A(I)
70 NEXT I
80 Y=A(1)
90 FOR I=2 TO 8
100 Y=Y*X+A(I)
110 NEXT I
120 PRINT 'Y=';Y
130 END
```

П. 26. Нахождение наибольшего значения. Составить программу условия задачи П. 13.

По условию задачи не требуется запоминать вычисляемые значения функции. Все переменные, используемые в программе, являются простыми. Исходные данные: В; промежуточные значения: У. Результат: У1.

Последовательность операторов в программе соответствует последовательности блоков на схеме рис. 13.

Программа выглядит следующим образом:

```
05 REM MAX
10 DATA ...
20 READ B
30 Y1=-1E10
40 FOR X=0 TO 4 STEP 0.1
50 Y=X*EXP(B*X-X*X)
60 IF Y>Y1 THEN 80
70 GOTO 90
80 Y1=Y
90 NEXT X
100 PRINT 'Y MAX=';Y1
110 END
```

П. 27. Нахождение наименьшего значения. Составить программу для условия задачи П. 14.

В программе следует выделить 40 ячеек памяти для запоминания значений массива. Последовательность операторов в программе соответствует последовательности блоков в схеме рис. 14. Ввод элементов массива осуществляется в цикле. Если изменить условие в условном операторе, то в программе будет исключен оператор безусловного перехода (вариант 2–фрагмент цикла). Замена переменных выполняется следующим образом:

условие	x_{min}	i_{min}
программа	XI	I1

Исходные данные: массив X. Результат: X1, I1.

Программа выглядит следующим образом:

Вариант 1

```
05 REM MIN
10 DIM X(40)
20 DATA...
30 FOR I=1 TO 40
```

Вариант 2

```
...
80 FOR I=2 TO 40
90 IF X(I)>=X1 THEN 120
100 I1=I
```

```
40 READ X(I)
50 NEXT I
60 X1=X(1)
70 I1=1
80 FOR I=2 TO 40
90 IF X(I)<X1 THEN 110
100 GOTO 130
110 X1=X(I)
120 I1=I
130 NEXT I
140 PRINT X1,I1
150 END
```

```
110 X1=X(I)
120 NEXT I
```

П. 28. Нахождение наименьшего значения. Составить программу для условия задачи П. 15.

По условию задачи все переменные в программе – простые. Последовательность операторов в программе соответствует блокам схемы на рис. 16. С целью исключения оператора безусловного перехода в программе условие в блоке *6* будет изменено на $y > y_{\min}$. Замена переменных выполняется следующим образом:

условие . . . . . $y_{\min}$
программа . . Y1

Исходные данные: A, C. Результат: Y1.
Программа будет выглядеть следующим образом:

```
05 REM MIN
10 DATA ...
20 READ A,C
30 Y1=1E10
40 FOR X=0 TO 4 STEP 0.2
50 Y=EXP(A*X+ABS(C)*X*X)
60 IF Y>=Y1 THEN 90
70 Y1=Y
80 NEXT X
90 PRINT 'Y MIN=';Y1
100 END
```

П. 29. Программирование алгоритмов со структурой вложенных циклов. Составить программу для условия задачи П. 6.

В схеме на рис. 6 не отражено выделение полей памяти под исходные данные массивов X и Y, а также под результат вычисления – матрицу Z. Это нужно сделать с помощью оператора DIM X(8), Y(10), Z(8, 10). Ввод исходных массивов X и Y можно осуществить раздельно, организуя для этого два отдельных цикла или поместив операторы ввода в самом вложенном цикле, до блока *4*. Задание конкретных значений элементов массивов осуществляется оператором DATA.

```
05 REM МАТРИЦА
10 DIM X(8),Y(10),Z(8,10)
20 DATA ...
30 DATA ...
40 FOR I=1 TO 8
50 READ X(I)
60 NEXT I
70 FOR J=1 TO 10
80 READ Y(J)
90 NEXT J
100 FOR I=1 TO 8
110 FOR J=1 TO 10
120 Z(I,J)=X(I)*Y(J)
130 PRINT Z(I,J)
140 NEXT J
150 NEXT I
160 END
```

П. 30. Составить программу для вычисления выражения

$$W=\sqrt{x^2+y^2+\sin xy}+\sqrt{y^2+z^2+\sin yz}+\sqrt{z^2+x^2+\sin zx}\ .$$

Это можно описать программой без использования оператора DEF (вариант 1), однако применение оператора DEF (вариант 2) сделает программу несколько компактнее. Для сравнения показаны два варианта программ:

Вариант 1

```
10 REM ФУНКЦИЯ W
20 DATA ...
30 READ X,Y,Z
40 W1=SQR(X*X+Y*Y+SIN(X*Y))
50 W2=SQR(Y*Y+Z*Z+SIN(Z*Y))
60 W3=SQR(Z*Z+X*X+SIN(Z*X))
70 W=W1+W2+W3
80 PRINT W
90 END
```

Вариант 2

```
10 REM ФУНКЦИЯ W С DEF
20 DATA ...
30 DEF FNW(A,B)=SQR(A*A+B*B+SIN(A*B))
40 READ X,Y,Z
50 W=FNW(X,Y)+FNW(Y,Z)+FNW(Z,X)
60 PRINT W
70 END
```

П. 31. Составить программу для вычисления числа сочетаний из n по m.

Число сочетаний из n по m определяется выражением

$$C_n^m=\frac{n!}{m!\,(n-m)!}$$

Эта задача может быть выполнена без использования подпрограммы. В этом случае нужно будет три раза повторить подготовку к накапливанию произведения, организацию цикла и накапливание произведения в цикле. Чтобы избежать повторения вычислений, нужно повторяющиеся операторы записать в подпрограмму.

В приведенной ниже подпрограмме это операторы с метками 160 – 190.

Программа с подпрограммой будет иметь вид:

```
20 DATA ...
30 READ N,M
40 K=N
50 GOSUB 160
60 N1=L
70 K=M
80 GOSUB 160
90 M1=L
100 K=N
110 GOSUB 160
120 M2=L
130 C=N1/M1/M2
140 PRINT C
145 STOP
160 L=1
170 FOR I=2 TO K
180 L=L*I
190 NEXT I
200 RETURN
210 END
```

При обращении к подпрограмме можно указывать метку 150 или 160. Операторы с метками 40, 70 и 100 задают фактические параметры, с которыми будут производиться вычисления в подпрограмме. При передаче управления из подпрограммы в программу необходимо запомнить вычисленные значения (операторы с метками 60, 90 и 120). С увеличением количества операторов в подпрограмме общее число операторов в программе будет уменьшаться по сравнению с программой без подпрограммы.

П. 32. Составить программу для определения среднего арифметического положительных элементов массивов A(50) и B(40), используя подпрограмму.

Как было показано выше, перед обращением к подпрограмме необходимо присвоить формальным параметрам значения соответствующих фактических параметров. В данной задаче формальными параметрами являются массив X и количество элементов этого массива N. Для того чтобы элементам формального массива X присвоить значения элементов фактического массива, например B, необходимо организовать цикл, который описывается операторами 130–160.

Это увеличивает объем программы и может свести на нет выгоду от использования подпрограммы. В целях уменьшения требуемого объема памяти формальный массив целесообразно хранить в том же месте, где и один из фактических массивов, например A. Тогда перед первым обращением к подпрограмме нет надобности переписывать элементы фактического массива A на место формального, а при втором обращении – осуществлять перезапись с помощью указанного выше цикла.

С учетом сказанного программа имеет вид:

```
20 A(50),B(40)
30 FOR I=1 TO 50
40 READ A(I)
50 NEXT I
60 DATA ...
70 FOR I=1 TO 40
80 READ B(I)
90 NEXT I
100 N=50
110 GOSUB 200
120 PRINT S
130 FOR I=1 TO 40
140 A(I)=B(I)
150 NEXT I
160 N=40
170 GOSUB 200
180 PRINT S
185 STOP
200 S=0
210 K=0
220 FOR I=1 TO N
230 IF A(I)<=0 THEN 260
240 S=S+A(I)
250 K=K+1
260 NEXT I
270 S=S/K
280 RETURN
290 END
```

ЗАДАЧИ И УПРАЖНЕНИЯ ПО ПРОГРАММИРОВАНИЮ ДЛЯ САМОСТОЯТЕЛЬНОГО РЕШЕНИЯ

40. Укажите допущенные ошибки в записи имен простых переменных:
1) A; 2) A1; 3) AI; 4) A6; 5) W5; 6) X; 7) Z9; 8) B; 9) β; 10) P4; 11) C10.

41. Укажите допущенные ошибки в записи элементов массивов:
1) A(1); 2) A (I); 3) AI(4); 4) B(3,5); 5) C25 (5); 6) D(I, I, I).

42. Записать операторы присваивания для вычисления значений арифметических выражений:

1) $t = \frac{x}{a} - \frac{1}{\pi}\lg(a+b)$;
2) $z = \frac{1}{x-1} + \sqrt{x+1}$;
3) $r = e^{z} - 1{,}6 \cdot 10^{3} \sin^{2} \frac{x}{\sqrt{ax}}$;
4) $p = 2{,}6x^{3} + 4{,}2x^{2} - 1{,}8x$;
5) $v = \frac{ax + b^{4}}{\sqrt[3]{(a-b)}}$;
6) $w = e^{\left(\frac{a+b}{c-d}\right)} + 10^{-4}t$;
7) $u = \sin^{3} \frac{x}{5} \cos x^{2} + e^{\sqrt{ax}}$;
8) $t = \frac{1}{y^{2}} \left(\frac{y}{10^{-3}}\right)^{x}$;
9) $y = \sqrt{1+x}\ \frac{\cos 2x}{1+\sqrt{x}}$;
10) $d = 2\sin^{3}(3,14 + z)$.

43. Записать фрагменты программ для вычисления следующих величин:

1) $w = \begin{cases} 1{,}25\, e^{-p}, \text{ если } p > 0, \\ 0{,}5\, e^{p}, \text{ если } p \leqslant 0; \end{cases}$

2) $t = \begin{cases} a \lg |x|, \text{ если } |x| < 1, \\ \sqrt{a + x^{2}}, \text{ если } |x| \geqslant 1; \end{cases}$

3) $z = \begin{cases} a + e^{bx}, \text{ если } a + b > 0, \\ b + \sin bx, \text{ если } a + b < 0, \\ x^{2}, \text{ если } a + b = 0; \end{cases}$

4) $f = \begin{cases} y^{3} - 0{,}5, \text{ если } y < 1{,}2 \text{ или } y > 2{,}5, \\ \sin y, \text{ если } 2{,}5 \geqslant y \geqslant 1{,}2; \end{cases}$

5) $s = \begin{cases} 0{,}5t^{2} - \sin t, \text{ если } t \geqslant 15{,}6, \\ e^{t+1}, \text{ если } 10{,}5 \leqslant t \leqslant 15{,}6, \\ 0{,}4\sin(t^{2}+1), \text{ если } t < 10{,}5; \end{cases}$

6) $q = \begin{cases} ce^{-px}, \text{ если } c^{2} - x \geqslant 0, \\ xe^{px}, \text{ если } c^{2} - x < 0. \end{cases}$

44. Записать фрагменты программы:
1) для размещения и ввода значений элементов массива T, состоящего из 20 элементов;
2) для вывода значений элементов массива T(20) с четными индексами по пять значений на строке;
3) для значений массива п. 2 с нечетными значениями индексов в компактной форме;
4) вывести значения элементов массива T(20) вместе с их номерами (индексами) располагая в каждой строке один элемент;

5) организовать цикл для вычисления значений функции $z = ax^2 + bx + c$, если x изменяется от 2 с шагом –0.2 до 0.

45. Составить программу для вычисления расстояний между двумя точками, лежащими на плоскости, по формуле

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \ .$$

Здесь $x_1, y_1; x_2, y_2$ – координаты точек.

46. Составить программу для вычисления по сторонам треугольника a, b, c высот треугольника:

$$h_a = \frac{2}{a}\sqrt{p(p-a)(p-b)(p-c)}; \quad h_b = \frac{2}{b}\sqrt{p(p-a)(p-b)(p-c)};$$
$$h_c = \frac{2}{c}\sqrt{p(p-a)(p-b)(p-c)};$$

медиан треугольника:

$$m_a = \frac{1}{2}\sqrt{2b^2 + 2c^2 - a^2}; \quad m_b = \frac{1}{2}\sqrt{2a^2 + 2c^2 - b^2}\ ;$$
$$m_c = \frac{1}{2}\sqrt{2b^2 + 2a^2 - c^2}\ ;$$

биссектрис треугольника:

$$\beta_a = \frac{2\sqrt{bcp(p-a)}}{b+c}\ ; \quad \beta_b = \frac{2\sqrt{acp(p-b)}}{a+c}\ ;$$
$$\beta_c = \frac{2\sqrt{abp\,(p-c)}}{b+a}\ ,$$

где $p = (a + b + c)/2$.

При разработке программы следует: исключить повторяющиеся вычисления, а замену имен переменных выполнить так:

условие	h_a	h_b	h_c	m_a	m_b	m_c	β_a	β_b	β_c
программа ...	H1	H2	H3	M1	M2	M3	B1	B2	B3

47. Составить программу для условия задачи 1. Замену имен переменных выполнить так:

условие	$d_н$	$d_в$	$d_{ср}$
программа	D1	D2	D3

48. Составить программу для условия задачи 3.

49. Составить программу для условия задачи 4.

50. Составить программу для условия задачи 6.

51. Составить программу для условия задачи 7.

Вместо γ использовать переменную G.

52. Составить программу для условия задачи 8.

• **53.** Составить программу для условия задачи 2.

Замену переменных производить не следует, так как все они записаны в соответствии с правилами записи переменных на языке БЕЙСИК.

• **54.** Составить программу для решения задачи 5. Замена переменных выполняется так:

условие	d_{a1}	d_{a2}	d_{f1}	d_{f2}
программа	D1	D2	D3	D4

Все остальные переменные, записанные строчными буквами, заменить на прописные.

● **55.** Составить программу для решения задачи 9. Замена переменных выполняется так:

условие	v_c	$x_{вл}$	P^*	k^*	γ_c	$\gamma_ф$	G_c	G_{oc}	$G_ф$	$F_ф$	τ
программа ..	VO	X1	P1	K1	HO	H1	G	GO	G1	F1	T

56. Составить программу для вычисления площади круга, который может быть задан радиусом r (признак $N = 1$), диаметром d (признак $N = 2$), длиной окружности c (признак $N = 3$).

$$s = \begin{cases} \pi r^2, \text{ если } N = 1, \\ \pi d^2/4, \text{ если } N = 2, \\ c^2/(4\pi), \text{ если } N = 3. \end{cases}$$

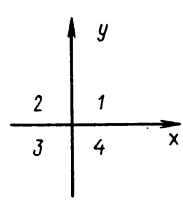
Исходное значение r, d и c в операторе READ следует обозначить через L, которое будет использовано в вычислениях s.

57. Составить программу для определения наибольшей площади фигур. Одна фигура является квадратом (площадь $s1$), а другая – кругом (площадь $s2$). Квадрат задан длиной стороны, а круг – радиусом. Результат вывести в виде ПЛОЩАДЬ КРУГА =или ПЛОЩАДЬ КВАДРАТА =.

58. Составить программу для определения, является ли треугольник, заданный координатами вершин $p_1(x_1, y_1)$, $p_2(x_2, y_2)$, $p_3(x_3, y_3)$, равносторонним, равнобедренным или разносторонним. Длины сторон треугольников между вершинами определять по формуле

$$L_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}.$$

59. Составить программу для определения, к какой четверти (квадранту) принадлежит точка T с координатами x и y ($x \neq 0$ и $y \neq 0$). Результатом является номер квадранта.



$$N = \begin{cases} 1, \text{если } x > 0 \text{ и } y > 0, \\ 2, \text{если } x < 0 \text{ и } y > 0, \\ 3, \text{если } x < 0 \text{ и } y < 0, \\ 4, \text{если } x > 0 \text{ и } y < 0. \end{cases}$$

60. Составить программу для условия задачи 12. Вместо w_g использовать переменную W1.

61. Составить программу для условия задачи 13. Вместо γ, μ и Re использовать переменные G1, M и R соответственно.

62. Составить программу для условия задачи 15. Вместо $F_п$ и $F_ц$ использовать переменные F1 и F2 соответственно.

• 63. Составить программу решения задачи 11. Вместо $F_{ткр}$ и $W_{тр}$ использовать переменные F1 и W1 соответственно.

• 64. Составить программу для решения задачи 10. Вместо W_r использовать W0.

• 65. Составить программу для решения задачи 14. Замена переменных выполняется так:

условие	. .	$t_{кон}$	$t_{ст}$	α	γ	μ	p_r	$p_{r\,ст}$	ϵ	λ	Re	Nu	β	G_r	g
программа	.	TO	T3	A1	C1	M0	P	PO	E	L	R	N	B1	G1	G

Табличные значения y_i сведем в массив Y, состоящий из 78 элементов. Значения индекса I изменяются от 0 до 77.

66. Составить программу для условия задачи 16. Вместо ρ и μ использовать переменные R и M соответственно.

67. Составить программу для условия задачи 21. Вместо γ, γ_1, γ_2 использовать переменные G, G1, G2.

68. Составить программу для условия задачи 22. Вместо F_v и W_0 использовать переменные F и W.

69. Составить программу для условия задачи 24. Вместо HB, D и $d_{ср}$ использовать переменные H, D1 и D2 соответственно.

70. Составить программу для вычисления значений функции $z = \sqrt{x_i + b_i}$, где $x = x_1, x_2, \ldots, x_{15}$, а b изменяется одновременно с i от b_0 с шагом h.

71. Составить программу для вычисления значений функции

$$y = \frac{e^{a_i x^2 + b_i x + c} - e^{2(a_i x^2 + b_i x + c)}}{a_i x^2 + bx + c},$$

где x изменяется одновременно с i от 0 до 1 с шагом 0.1, а $i = 1, 2, 3, \ldots 11$. Повторяющиеся вычисления исключить.

72. Составить программу для вычисления значений функции $t = \sqrt{\frac{a_i + b_i + c_i}{i}}$, где $a = a_1, a_2, \ldots, a_{15}$; b изменяется от 0 с шагом 0.3, а c изменяется от 0 с шагом 0.5.

73. Составить программу для записи элементов массива $x_1, x_2, \ldots, x_{20}$ в массив Y в обратном порядке. Результат вывести на экран дисплея.

74. Составить программу для записи подряд элементов массива $a_1, a_2, \ldots, a_{50}$, имеющих четные индексы, в массив W, а нечетные – в массив T. Результат вывести на экран дисплея.

75. Составить программу для записи подряд положительных элементов массива $x_1, x_2, \ldots, x_{100}$ в массив Y, а в массив Z – отрицательных.

76. Составить программу для вычисления и запоминания в массиве Y значений функции $y_i = a\,e^{x_i}$, где $i = 1, 2, \ldots, 60$.

77. Составить программу для вычисления и запоминания в массиве Z значений функции $z = a\,e^x$, где x изменяется от 0 до 1 с шагом 0.1.

78. Составить программу для вычисления среднего арифметического значения элементов массива $a_1, a_2, \ldots, a_n$. Исходные данные: массив A. Результат: S.

79. Составить программу для вычисления среднего арифметического значения положительных элементов массива $c_1, c_2, \ldots, c_n$ (полагая, что в массиве есть положительные элементы). В программе потребуется не только суммировать положительные элементы массива, но и подсчитать их количество. Для этого перед циклом нужно задать начальное значение $M = 0$.

Исходные данные: массив C; промежуточные значения: сумма S, количество положительных элементов M. Результат: S1.

80. Составить программу для условия задачи 29. Исходные данные: N, массив R. Результат: R0.

81. Составить программу для условия задачи 31. Исходные данные: N. Результат: M.

82. Составить программу для условия задачи 32. Исходные данные: N, M. Результат: C.

83. Составить программу для вычисления $z = 1 + x + \frac{x^2}{2} + \ldots + \frac{x^{20}}{20}$.

84. Составить программу для вычисления произведения элементов массива $a_1, a_2, \ldots, a_{70}$, кратных 5.

85. Составить программу для вычисления среднего геометрического положительных элементов массива $x_1, x_2, \ldots, x_{40}$.

86. Вычислить сумму положительных и сумму отрицательных элементов массива $a_1, a_2, \ldots, a_{50}$.

87. Составить программу для вычисления суммы членов ряда

$$y = \frac{1}{2 \cdot 3} + \frac{2}{3 \cdot 4} + \ldots + \frac{n}{(n+1)(n+2)} + \ldots$$ с точностью до члена ряда, меньшего 10^{-4}.

88. Составить программу для вычисления суммы членов ряда

$$z = \frac{1}{1 \cdot 3} + \frac{1}{3 \cdot 5} + \frac{1}{5 \cdot 7} + \ldots + \frac{1}{(2n-1)(2n+1)} + \ldots$$ с точностью до члена ряда, меньшего ϵ.

89. Составить программу для вычисления $y = x - \frac{x^2}{2!} + \frac{x^4}{4!} - \ldots + (-1)^i \frac{x^{2i}}{(2i)!} + \ldots$ с точностью до члена ряда, меньшего 10^{-5}.

90. Составить программу для вычисления $y = 1 + \frac{\sin x}{1} + \frac{\sin 2x}{2} + \ldots + \frac{\sin nx}{n} + \ldots$ с точностью до члена ряда, меньшего 10^{-3}.

91. Составить программу для вычисления многочлена $z = 2x^8 - x^6 + 4x^5 - 5x^2 + 6x + 1$. Схема программы аналогична задаче П. 25. Коэффициенты многочлена необходимо задать массивом.

Исходные данные: X, массив A. Результат: Z.

92. Составить программу для вычисления многочлена $s = 1x^8 + 2x^7 + 3x^6 + 4x^5 + 5x^4 + 6x^3 + 7x^2 + 8x + 9$. Так как коэффициенты многочлена — числа натурального ряда, то целесообразно их вычислять в программе. Исходные данные: X. Результат: S.

93. Составить программу для вычисления многочлена $y = a_1 x^n + a_2 x^{n-1} + \ldots + a_n x + a_{n+1}$. Коэффициент многочлена свести в массив.

94. Составить программу для нахождения наибольшего значения $(x_i + y_i)$ элементов массивов $x_1, x_2, \ldots, x_n$ и $y_1, y_2, \ldots, y_n$. Если начальное значение задать оператором присваивания $(x_1 + y_1)$, то потребуется ввод массивов выполнить отдельно от вычислений. При задании начального значения, равного -10^{10}, ввод массивов можно выполнять в основном цикле. Чтобы дважды не вычислять $(x_i + y_i)$, вводится промежуточная переменная $T = x_i + y_i$. Исходные данные: массивы X и Y; промежуточные переменные: T. Результат: Z.

95. Составить программу для нахождения наименьшего из положительных элементов массива $a_1, a_2, \ldots, a_n$. В качестве начального значения наименьшего A1 использовать первый элемент массива.

96. Составить программу для нахождения наименьшего значения функции $y = ax^3 + bx^2 + cx + d$ и значения аргумента, при котором оно получено. Значение аргумента x изменяется от 0 до 10 с шагом 0.1.

97. Составить программу для нахождения наибольшего и наименьшего элементов массива $a_1, a_2, \ldots, a_{80}$.

98. Составить программу для нахождения наибольшего значения функции $y = a \sin(ax + \varphi)$ при изменении аргумента x от 0 до $\pi/2$ с шагом 0.1. Функция в указанном диапазоне имеет один максимум.

99. Составить программу для нахождения наименьшего значения и его номера среди элементов массива $x_1, x_2, \ldots, x_{50}$, стоящих на четных местах.

100. Составить программу для вычисления значения функции по формуле $y_{i+1} = \frac{3}{2} y_i - \frac{1}{2} x y_i^3$ с точностью ϵ, принять $y_0 = 2(2 - \sqrt{2})$.

101. Составить программу для получения матрицы B (6, 5) из матрицы A (5, 6), поменяв местами строки и столбцы.

102. Составить программу для вычисления функции $r = \sum_{j=1}^{10} \sum_{k=1}^{n} \frac{\sin^k(x_j)}{k}$, где x_j – элементы массива $x_1, x_2, \ldots, x_{10}$; $k = 1, 2, \ldots, n$.

103. Составить программу для определения количества положительных элементов каждого столбца матрицы A (10, 20) и запомнить их в массиве M.

104. Составить программу для нахождения наибольшего значения элемента матрицы D (20, 30) и номера строки и столбца, в которых он расположен.

105. Составить программу для вычисления средних арифметических положительных значений элементов каждого столбца матрицы T (n, m) при условии, что в каждом столбце есть хотя бы один положительный элемент.

106. Составить программу для условия задачи 30.

107. Составить программу для условия задачи 37. Замену переменных выполнить так:

условие	$\Delta q_т$	$V_т$	ΔV_0	$V_{0\,min}$	$V_{0\,max}$	$V_и$
программа ...	Q0	W	V0	V1	V2	V3

• **108.** Составить программу для решения задачи 17.

● **109.** Составить программу для решения задачи 18. Замена переменных выполняется так:

условие . . . . . α_1 α_2 λ_1 λ_2
программа . . . A1 A2 L1 L2

110. Составить программу для решения задачи 20.

● **111.** Составить программу решения задачи 19. Табличные значения w_i' сведены в массив W, состоящий из 21 элемента. Принять следующие обозначения: V0 – предыдущее значение скорости сушки, V – текущее значение скорости сушки, $k = \Delta V/V$, обозначить $w'_{кр}$ как W0, а $t_{кр}$ как T0; ΔT считать равным 1, $\Delta w'$ определить как W(I – 1) – W(I), так как значения w_i' сведены в массив.

● **112.** Составить программу для решения задачи 23. Вместо Δt, V_c и $I_м$ использовать переменные H, V и J соответственно.

● **113.** Составить программу для решения задачи 25. Средние значения $\overline{T}$, $\overline{A}$, $\overline{m}$ обозначить T0, A0 и M0 соответственно. Табличные значения m_i, A_i и T_i свести в массивы M, A, T, состоящие из 100 элементов каждый. Тем самым введено ограничение на значение N, которое не должно превосходить 100.

● **114.** Составить программу для решения задачи 26. Ввести обозначение X0 вместо $\overline{X}$. Значения таблицы испытаний x_i свести в массив X, состоящий из 100 элементов. Реальное количество испытаний N не должно превышать 100. Поскольку индексация элементов массива начинается не с нуля, в операторе DIM следует указывать последнее значение индекса, равное 100.

● **115.** Составить программу для решения задачи 27. Ввести обозначения X0 вместо X_m, Y0 вместо Y_m и Z0 вместо Z_m, M0 обозначить Σm_i. Массы m_i и координаты точек x_i, y_i, z_i свести соответственно в массивы M, X, Y, каждый из которых состоит из 100 элементов. Реальное количество точек N не должно превышать 100.

▲ ● **116.** Составить программу для решения задачи 28. Замена переменных выполняется так:

условие ..... t_1 t_2 t
программа..... X1 X2 X

Реальное количество N элементов массива не может быть больше 20.

▲ ● **117.** Составить программу для решения задачи 33. Замена переменных: X вместо γ; Y вместо η; Y0 вместо η_{max}; Q0, I0 вместо Q и I, при которых η равно η_{max}. Табличные значения H_i и N_i свести в массивы H и N, содержащие по 11 элементов.

▲ ● **118.** Составить программу для решения задачи 34. Произвести замену переменных: Q0 вместо q; Q1 вместо q_m; Q3 вместо Q_{max}; Q2 вместо q_{CH4}; T4 вместо $T_{уч}$; T0 вместо t; Q4 вместо $q_{уч}$. Вычисленные значения записываются в массив, состоящий из трех элементов.

▲ ● **119.** Составить программу решения задачи 35. Замена переменных: L вместо ΔL, I0 вместо I_x; значения расстояний от продольной оси до

борта свести в массив Y, состоящий из $(n + 1)$ элементов. В операторе DIM размер массива можно указывать только числом, поэтому описать массив Y (100), за счет чего N не может иметь значение, большее 100.

▲ ● **120.** Составить программу решения задачи 36. Замена переменных: V0 вместо ΔV. Табличные значения p_i свести в массив P, состоящий из $2n$ элементов. В операторе DIM размер массива можно задавать только числом, поэтому описать массив P как состоящий из 100 элементов, тогда N не может быть больше 50; так как индексация начинается не с нуля, то массив будет описываться как DIM P(100).

▲ ● **121.** Составить программу для решения задачи 38. Замена переменных: V0 вместо $V_{\min}$; V1 вместо $V_{\max}$; V2 вместо ΔV; T2 вместо ΔT_k.

▲ ● **122.** Составить программу решения задачи 39. Замена переменных: X0 вместо $\bar{X}_G$ и X_G; X1 вместо $\bar{X}_p$ и X_p; P0 вместо P; G0 вместо $\bar{G}$. Табличные значения G_i, P_i и X_i свести в массивы G, P, X, состоящие из n элементов каждый. Полагаем, что n не превосходит 100.

123. Составить программу для вычисления выражений $c = \sqrt{\frac{a^4 - 1}{a^3 + 2}}$ и $d = \frac{b^4 - 1}{b^3 + 2} \frac{t^4 - 1}{t^3 + 2}$, используя оператор DEF.

124. Составить программу для вычисления выражений

$$w = \frac{\sqrt{sz^3 + qz^2 + qz + t}}{1 + e^{sz^3} + qz} \quad \text{и} \quad f = \frac{\alpha t^3 + \beta t^2 + \gamma t + r}{r^3 + 2r - v},$$

используя оператор DEF.

125. Составить программу для вычисления выражений $z = \frac{a_{\max} + b_{\max} + c_{\max}}{3}$, где $a_{\max}$, $b_{\max}$, $c_{\max}$ – наибольшие элементы массивов A(10), B(15), C(18) соответственно. Нахождение наибольших элементов выполнить в подпрограмме.

126. Составить программу для вычисления $z = \frac{\log_2 x + \log_2 y}{\log_{(b+2)}(x+y)}$, используя оператор DEF.

▲ **127.** Составить программу для определения группы учащихся, имеющей наибольший средний балл по результатам экзаменов. Результаты экзаменов сведены в матрицы A(25,5), B(23,5), C(24,5). Средний балл группы определять в подпрограмме.

▲ **128.** Составить программу для определения количества учащихся, имеющих оценки "отлично" n_5 и оценки "отлично" и "хорошо" n_{45}. Оценки для каждой группы сведены в массивы A(25,9), B(28,9), C(23,9). Для определения n_5 и n_{45} использовать подпрограмму.

ОТВЕТЫ И ПОЯСНЕНИЯ

1. *Математическая формулировка задачи* сводится к выполнению расчетов по формулам: $l = F(\pi d_{ср} m)$; $n = l/l_1$.

Расчетный диаметр трубы $d_{ср}$ вычисляется по формуле $d_{ср} = (d_н + d_в)/2$ при условии, что α_1 (коэффициент теплопередачи от горячего теплоносителя) и α_2 (коэффициент теплоотдачи от стенки к холодному теплоносителю) одного порядка; $d_н$ – диаметр наружной трубы; $d_в$ – диаметр внутренней трубы. Метод вычисления определяется приведенными формулами.

Алгоритм решения задачи

1. Ввод исходных данных: F, $d_н$, $d_в$, m, l_1. 2. Вычисление $d_{ср}$. 3. Вычисление l. 4. Вычисление n. 5. Вывод результатов l и n.

Схема программы, описывающая алгоритм, представлена на рис. 18.

2. *Математическая формулировка задачи*

Скорость движения раствора определяется по формуле $V = Q/F$, где Q – производительность насоса; F – площадь поперечного сечения кольцевого пространства между стенками скважины и бурильными трубами.

Площадь сечения скважины $F_1 = \pi D_1^2/4$; площадь сечения бурильной трубы $F_2 = \pi D_2^2/4$. Тогда $F = F_1 - F_2$.

Алгоритм решения задачи

1. Ввод исходных данных D_1, D_2, Q. 2. Вычисление F_1, F_2, F и V. 3. Вывод результата V.

Схема программы, описывающая алгоритм, представлена на рис. 19.

3. *Математическая формулировка задачи*

Расчетная масса состава определяется по формуле

$$Q = \frac{F - P(W_0 + i_0)}{W_1 + i_0},$$

где W_1 – удельное сопротивление четырехосных вагонов, вычисляемое по формуле

$$W_1 = 0{,}7 + \frac{8{,}0 + 0{,}1V + 0{,}0025V^2}{q/4};$$

$i_0 = i + 700/R$ – удельное дополнительное сопротивление в зависимости от подъема и кривизны пути.

Алгоритм решения задачи

1. Ввод исходных данных F, V, W_0, P, q, R, i. 2. Вычисление W_1, i_0, Q. 3. Вывод результата Q.

Схема программы, описывающая алгоритм, представлена на рис. 20.

4. *Математическая формулировка задачи*.

При прохождении тока $I_1 + I_2$ можно записать $t_1 = m/[k(I_1 + I_2)]$;

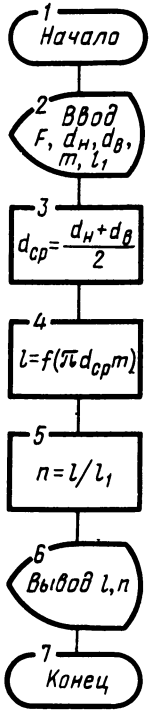


Рис. 18

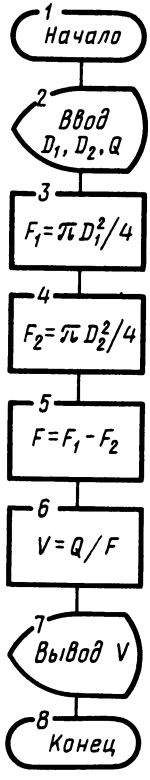


Рис. 19

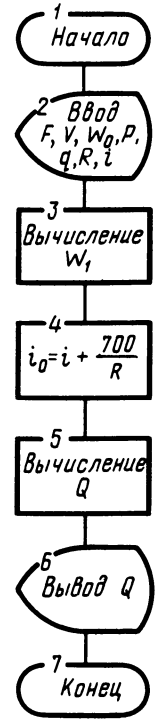


Рис. 20

для токов I_1 и I_2 получим $t_2 = \frac{m/2}{kI_1} + \frac{m/2}{kI_2}$. Требуется найти $t = m/(kI_1)$.

Из выражения для t_2 получим $t = 2t_2 - m/(kI_2)$, из выражения для t_1 получим $t_1 kI_1/m + t_1 kI_2/m = 1$, откуда $kI_2/m = 1/t_1 - kI_1/m = 1/t_1 - 1/t$. Подставляя kI_2/m в уравнение для t, получим $t = 2t_2 - 1/(\frac{1}{t_1} - \frac{1}{t})$ или $t^2 - 2t_2 t + 2t_1 t_2 = 0$.

Корни уравнения определяются по формуле

$$\tau_{1,2} = t_2 \pm \sqrt{t_2^2 - 2t_1 t_2}.$$

Оба корня действительные, так как подкоренное выражение больше нуля. В этом нетрудно убедиться, подставляя вместо t_1 и t_2 соответствующие им выражения.

Алгоритм решения задачи

1. Ввод исходных данных t_1, t_2. 2. Вычисление t. 3. Вывод результата t.

Схема программы, описывающая алгоритм, представлена на рис. 21.

5. *Математическая формулировка задачи*

Межосевое расстояние

$$a = m\left(\frac{z_1 + z_2}{2}\right), \qquad (1)$$

где z_1, z_2 – число зубьев первого и второго колес. Диаметр делительной окружности колеса $d = mz$, а передаточное число

$$n = z_2/z_1 \qquad (2)$$

Решая совместно уравнения (1) и (2), получаем: $z_1 = 2a/[m(n + 1)]$ и $z_2 = 2an/[m(n + 1)]$.

Диаметры делительных окружностей равны: $d_1 = mz_1$ и $d_2 = mz_2$. Диаметры окружности выступов $d_{a_1} = m(z_1 + 2)$ и $d_{a_2} = m(z_2 + 2)$, а диаметры впадин $d_{f_1} = m(z_1 - 2{,}5)$ и $d_{f_2} = m(z_2 - 2{,}5)$.

Алгоритм решения задачи

1. Ввод исходных данных a, n, m. 2. Вычисление z_1, z_2, d_{a_1}, d_{a_2}, d_{f_1}, d_{f_2}. 3. Вывод результатов d_{a_1}, d_{a_2}, d_{f_1}, d_{f_2}.

Схема программы, описывающая алгоритм, представлена на рис. 22.

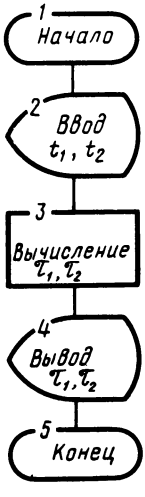


Рис. 21

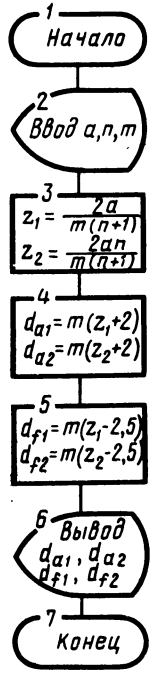


Рис. 22

6. *Математическая формулировка задачи*

Зависимость объема фильтрата V от времени фильтрации t при постоянном гидравлическом сопротивлении определяется уравнением

$$V^2 + 2Vc = kt \ .$$

Для определения V необходимо решить это квадратное уравнение. Корни квадратного уравнения $V_{1,\ 2} = -c \pm \sqrt{c^2 + kt}$. Поскольку $c > 0$ и $k > 0$ и объем фильтрата не может быть отрицательным, приемлем только один корень уравнения $V = -c + \sqrt{c^2 + kt}$.

Алгоритм решения задачи

1. Ввод исходных данных t, k, c. 2. Вычисление V. 3. Вывод результата V.

Схема программы, описывающая алгоритм, представлена на рис. 23.

7. *Математическая формулировка задачи*

Уравнение материального баланса процесса фильтрации $G_{\text{сусп}} = G_{\text{ф}} + G_{\text{вл}}$, где $G_{\text{сусп}}$ – масса суспензии; $G_{\text{ф}}$ – масса фильтрата; $G_{\text{вл}}$ – масса влажного осадка. Разделив обе части уравнения на $G_{\text{сух}}$ – массу сухого вещества, содержащегося в суспензии, получим $G_{\text{сусп}}/G_{\text{сух}} = G_{\text{ф}}/G_{\text{сух}} + G_{\text{вл}}/G_{\text{сух}}$.

Поскольку $G_{\text{ф}} = V_{\text{ф}}\gamma$, где $V_{\text{ф}}$ – объем фильтрата, то, обозначив $m = G_{\text{вл}}/G_{\text{сух}}$ и $x = G_{\text{сух}}/G_{\text{сусп}}$, получим $1/x = \gamma/c + m$, откуда $c = \gamma x/(1 - mx)$.

Алгоритм решения задачи

1. Ввод исходных данных γ, x, m. 2. Вычисление C. 3. Вывод результата C.

Схема программы, описывающая алгоритм, представлена на рис. 24.

8. *Математическая формулировка задачи*

Подставляя в уравнение фильтрации $V^2 + 2Vc = kt$ значения V_1, t_1 и V_2, t_2, получим систему уравнения

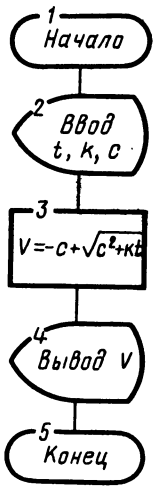


Рис. 23

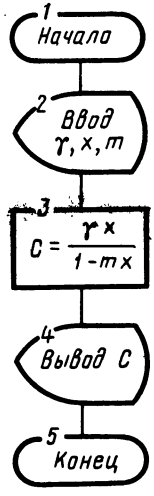


Рис. 24

$$V_1^2 + 2V_1 c = kt_1;$$
$$V_2^2 + 2V_2 c = kt_2 .$$

Решая систему уравнений относительно c и k, получим

$$c = \frac{t_1 V_2^2 - t_2 V_1^2}{2(V_1 t_2 - V_2 t_1)}, \quad k = \frac{(V_2 - V_1) V_1 V_2}{t_2 V_1 - t_1 V_2} .$$

Из уравнения фильтрации находится продолжительность фильтрации

$$t = (V_0^2 + 2V_0 c)/k .$$

Алгоритм решения задачи

1. Ввод исходных данных V_0, V_1, V_2, t_1, t_2. 2. Вычисление c, k, t. 3. Вывод результатов c, k, t.

Схема программы, описывающая алгоритм, представлена на рис. 25.

9. *Математическая формулировка задачи*

Пересчитаем константу k для вакуума P на заводе: $k/k^* = P/P^*$, откуда $k = k^* P/P^*$.

Удельная производительность зоны фильтрации определяется из уравнения $V^2 + 2cV = k\tau$, откуда $V = -c + \sqrt{c^2 + k\tau}$. Вторым корнем уравнения

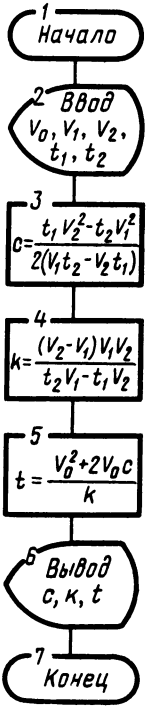


Рис. 25

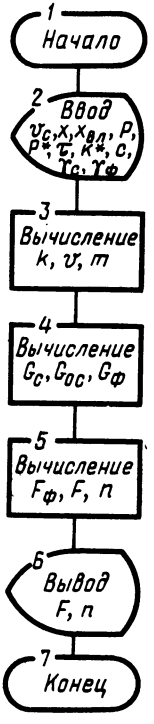


Рис. 26

пренебрегаем, так как он дает отрицательное значение V, что не имеет физического смысла. Удельная производительность зоны фильтрации в секунду равна V/τ.

Отношение влажного осадка к сухому $m = 1/(1 - x_{вл}/100)$. Массовая доля твердой фазы в суспензии $x/100$; расход суспензии $G_c = \gamma_c v_c$; масса влажного осадка $G_{ос} = G_c x m/100$; масса фильтрата $G_ф = G_c - C_{ос}$; объемный расход фильтрата $G_ф/\gamma_ф$; производительность по фильтрату $G_ф/(3600\gamma_ф)$. Следовательно, необходимая поверхность в зоне фильтрации $F_ф = G_ф\tau/(3600\gamma_ф V)$. Так как в барабанных вакуум-фильтрах поверхность зоны фильтрации составляет около 35 % общей поверхности F, то $F = F_ф/0{,}35$. По значению F из каталога выбирают ближайший тип барабана.

Число оборотов вакуум-фильтра в минуту, необходимое для обеспечения заданного времени фильтрации τ, определяется из выражения $n = 60 \cdot 0{,}35/\tau$.

Алгоритм решения задачи

1. Ввод исходных данных V_c, x, $x_{вл}$, P, P^*, τ, k^*, c, γ_c, $\gamma_ф$. 2. Вычисление k, V, m, G_c, $G_{ос}$, $G_ф$, $F_ф$, F, n. 3. Вывод результатов F и n.

Схема программы, описывающая алгоритм, представлена на рис. 26.

10. *Математическая формулировка задачи*

Основное сопротивление электровозу при движении под током определяется формулой $W_1 = (1{,}9 + 0{,}01V + 0{,}0003V^2)P$. Основное сопротивление груженым четырехосным вагонам на подшипниках скольжения определяется формулой

$$W_2 = 0{,}7 + (8 + 0{,}1V + 0{,}0025V^2)/q_0 ,$$

где q_0 — средняя нагрузка от каждой оси на рельсы, которая может быть определена как $q_0 = Q/(4n)$.

Сопротивление за счет уклона и кривизны пути определяется формулой

$$W_3 = (P + Q)(i + W_r),$$

где

$$W_r = \begin{cases} \frac{700}{R}, \text{ если } L \leqslant S, \\ \frac{700}{R} - \frac{S}{L}, \text{ если } L > S. \end{cases}$$

Полное сопротивление поезду $W = W_1 + W_2 + W_3$.

Алгоритм решения задачи

1. Ввод исходных P, Q, n, V, R, S, L, i. 2. Вычисление W_1, q_0, W_2. 3. Проверка условия: если $L \leqslant S$, то вычисление $W_r = 700/R$, в противном случае $W_r = \frac{700}{R} - \frac{S}{L}$. 4. Вычисление W_3, W. 5. Вывод W.

Схема программы, описывающая алгоритм, представлена на рис. 27.

11. *Математическая формулировка задачи*

Средняя нагрузка, передающаяся от оси на рельсы, $q_0 = Q/k$. Удельное сопротивление трогания для подшипников скольжения $W_{тр} = 142/(q_0 + 7)$, а для роликовых $W_{тр} = 28/(q_0 + 7)$.

Масса состава, при которой электровоз может сдвинуть состав с мес-

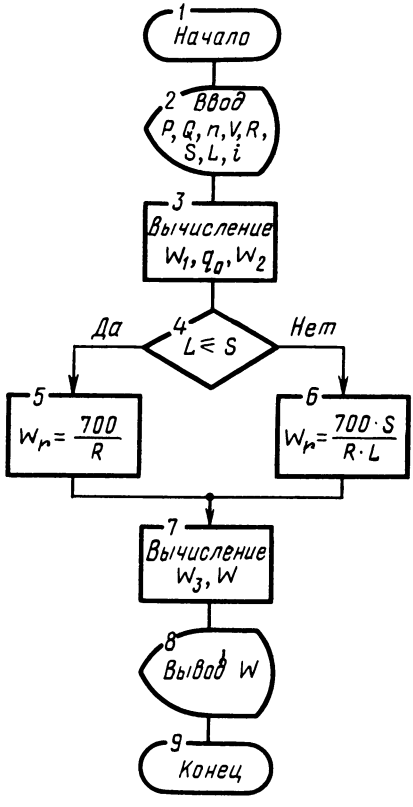


Рис. 27

та, определяется формулой $Q_{тр} = F_{k\,тр}/(W_{тр} + i) - P$. Откуда для состава массой $Q{:}i = F_{k\,тр}/(Q + P) - W_{тр}$, где $F_{k\,тр}$ — сила тяги, которую может развивать электровоз в момент трогания (зависит от типа электровоза); P — масса электровоза. Введем вспомогательный параметр, определяющий тип подшипников: $n_1 = 0$ — для подшипников скольжения, $n_1 = 1$ — для роликовых подшипников.

Алгоритм решения задачи

1. Ввод исходных данных $n_1, Q, k, P, F_{k\,тр}$. 2. Вычисление q_0. 3. Проверка условия $n_1 = 0$; если оно выполняется, то вычисление $W_{тр} = 142/(q_0 + 7)$, в противном случае вычисление $W_{тр} = 28/(q_0 + 7)$. 4. Вычисление i. 5. Вывод результата i.

Схема программы, описывающая алгоритм, представлена на рис. 28.

12. *Математическая формулировка задачи*

Основное удельное сопротивление четырех- и шестиосных вагонов определяется формулой

$$W_0 = 0{,}7 + (8{,}0 + 0{,}1\,V + 0{,}0025\,V^2)/q_0\,,$$

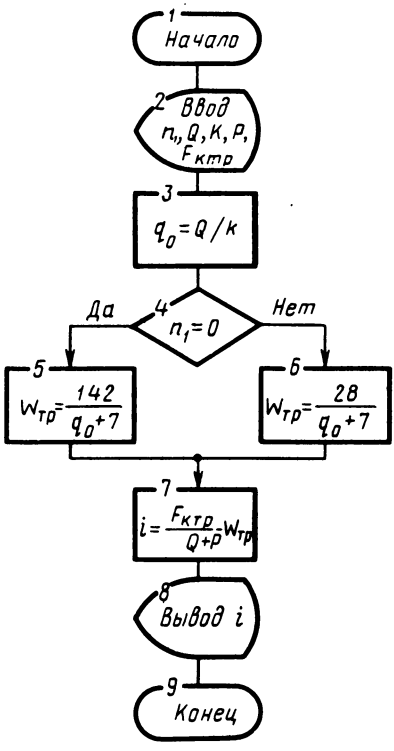


Рис. 28

где V — скорость движения поезда; q_0 — средняя нагрузка от каждой оси на рельсы.

Решение задачи сводится к нахождению корней квадратного уравнения:

$$0{,}0025\,V^2 + 0{,}1\,V + 8{,}0 - (W_g - 0{,}7)\,q_0 = 0,$$

или

$$V^2 + 40\,V + 3200 - 400\,(W_g - 0{,}7)\,q_0 = 0.$$

Корни квадратного уравнения определяются по формуле

$$V_{1,2} = -20 \pm \sqrt{400 - 3200 + 400\,(W_g - 0{,}7)\,q_0}\,.$$

При определенных (неправильно выбранных) значениях W_g и q_0 может оказаться, что подкоренное выражение будет отрицательным; ЭВМ не может вычислять квадратный корень из отрицательного числа. Поэтому в алгоритме необходимо предусмотреть эту опасность, тем более, что физичес-

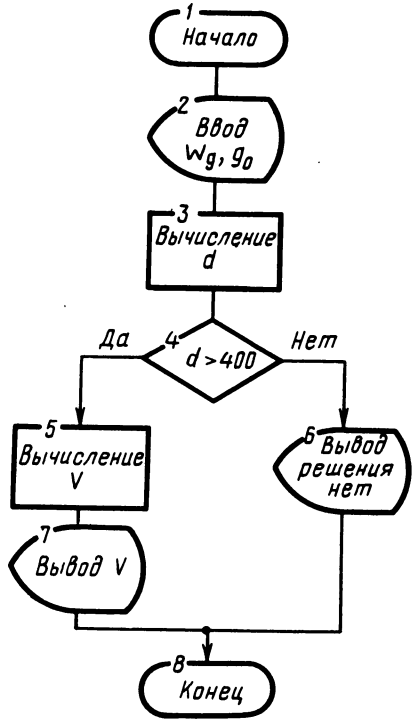


Рис. 29

кого смысла комплексные корни не имеют. С другой стороны, из двух действительных корней имеет смысл только положительный корень. Тогда процесс вычисления корня уравнения можно описать следующей формулой:

$$V=\begin{cases}-20+\sqrt{-2800+400(W_g-0{,}7)\,q_0}, \text{ если } -2800+400(W_g-0{,}7)\,q_0>400,\\ \text{решение не имеет смысла, если } -2800+400(W_g-0{,}7)\,q_0\leqslant 400.\end{cases}$$

Алгоритм решения задачи

1. Ввод исходных данных W_g, q_0. 2. Вычисление $d=-2800+400(W_g-0{,}7)\,q_0$. 3. Проверка условия: если $d>400$, то вычисление V, вывод результата V, в противном случае ($d\leqslant 400$) вывод сообщения "Решения нет".

Схема программы, описывающая алгоритм, представлена на рис. 29.

13. *Математическая формулировка задачи*

Для решения задачи необходимо определить коэффициент Рейнольдса по формуле $\mathrm{Re}=dW\gamma/(\mu g)$, где d – диаметр трубопровода, W – скорость жидкости, g – ускорение свободного падения, равное 9,81 м/с^2.

Скорость жидкости находится из уравнения расхода $W=G/(f\gamma)$, где $f=\frac{\pi((d_2-h_2)^2-d_1^2)}{4}$ – площадь сечения трубопровода.

Откуда $W = \dfrac{4G}{\gamma\pi((d_2 - h_2)^2 - d_1^2}$.

Эквивалентный диаметр трубопровода кольцевого сечения $d = d_2 - h_2 - d_1$. Режим течения ламинарный, если $Re < R_0$, и турбулентный, если $Re \geqslant R_0$, где R_0 – константа, зависящая от материала трубы и шероховатости стенок.

Алгоритм решения задачи

1. Ввод исходных данных d_1, d_2, h_1, h_2, G, γ, μ, R_0. 2. Вычисление f, d, W, Re. 3. Проверка условия: если $Re < R_0$, то вывод "ламинарный", в противном случае вывод "турбулентный".

Схема программы, описывающая алгоритм, представлена на рис. 30.

14. *Математическая формулировка задачи*

Средняя температура воды $t_ж = (t_{нач} + t_{кон})/2$. Определяется режим течения: $Re = wd\gamma/(\mu g)$, где d – внутренний диаметр трубы; γ – плотность жидкости; μ – динамический коэффициент вязкости жидкости; g – ускорение свободного падения. В зависимости от значения Re поток: турбулентный, если $Re \geqslant 10\,000$, переходный, если $2300 < Re < 10\,000$; ламинарный, если $Re < 2300$.

Для турбулентного потока критерий Нуссельта $Nu = 0{,}021\epsilon Re^{0,8} Pr^{0,43}(Pr/Pr_{ст})^{0,25}$ (1), где Pr и $Pr_{ст}$ – критерий Прандтля для потока (выбирается из таблицы); ϵ – коэффициент, зависящий от соотношения длины трубы к ее диаметру (выбирается из таблицы).

Для ламинарного потока критерий $Nu = 0{,}15\, Re^{0,33} Pr^{0,43} G_r^{0,1}(Pr/Pr_{ст})^{0,25}$... ... (2) $G_r = d^3\gamma^2\beta\Delta t/(\mu^2 g)$, где $\Delta t = t_{ст} - t_{кон}$; β – коэффициент объемного расширения, зависящий от температуры,

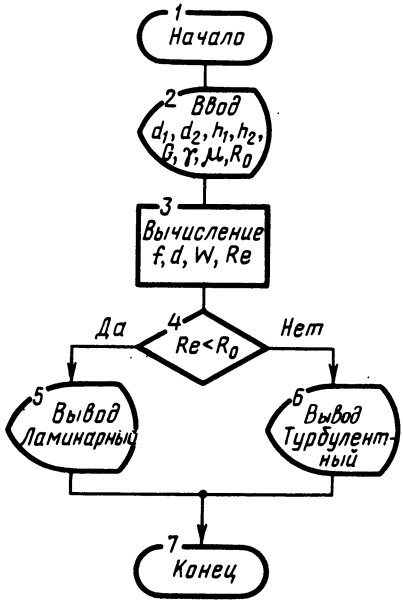


Рис. 30

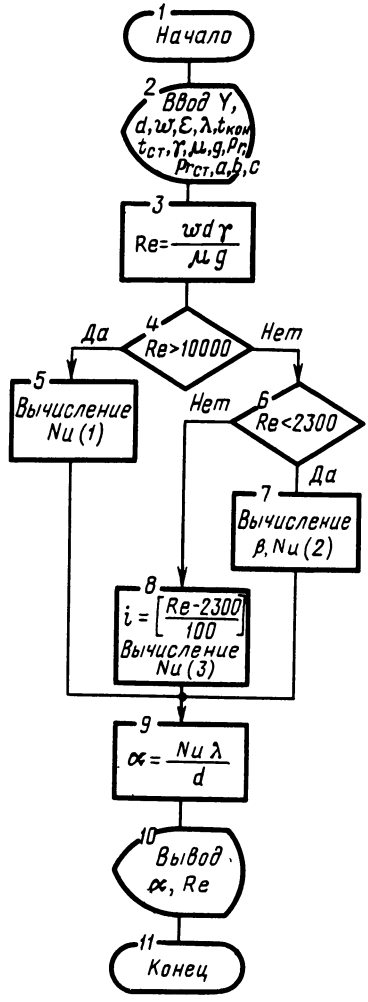


Рис. 31

$$\beta = \frac{a(t_1 - t_2) + b(t_1^2 - t_2^2) + c(t_1^3 - t_2^3)}{t_1 - t_2}.$$

Значения коэффициентов a, b, c выбираются из справочника. Принять $t_1 = t_{кон} + 1$; $t_2 = t_{кон} - 1$.

Для переходного режима зависимость $\frac{Nu}{Pr^{0,43}(Pr/Pr_{ст})^{0,25}}$ от Re находится из графика. Задать значения этой функции с помощью таблицы при изменении аргумента Re от 2300 до 10 000 с шагом 100. Зная значения Re из таблицы, найти значения $y_i = \frac{Nu}{Pr^{0,43}(Pr/Pr_{ст})^{0,25}}$. Откуда $Nu = y_i Pr^{0,43}(Pr/Pr_{ст})^{0,25}$. (3) Для определения y_i необходимо знать, какой

номер имеет значение в таблице для заданного значения Re. Поскольку шаг изменения аргумента Re равен 100, то в таблице будет [(10 000 – – 2300)/100] + 1 элементов. Если аргумент имеет значение Re*, то ему будет соответствовать функция с порядковым номером i в таблице, вычисляемым по следующей формуле: $i = [(\mathrm{Re}^* - 2300)/100]$. Квадратные скобки означают, что берется целая часть числа.

Далее для любого потока коэффициент теплопроводности $\alpha = \mathrm{Nu}\lambda/d$, где λ – теплопроводность жидкости при $t_ж$ выбирается из справочника.

Алгоритм решения задачи

1. Ввод исходных данных d, w, ϵ, λ, $t_{кон}$, $t_{ст}$, γ, μ, g, Pr, $\mathrm{Pr}_{ст}$, a, b, c и массив значений Y. 2. Вычисление Re. 3. Проверка условия: если $\mathrm{Re} > 10\,000$, то вычисление Nu по формуле (1), в противном случае проверка условия: если $\mathrm{Re} < 2300$, то вычисление β и Nu по формуле (2), в противном случае вычисление i и Nu по формуле (3). 4. Вычисление α. 5. Вывод результатов α и Re.

Схема программы, описывающая алгоритм, представлена на рис. 31.

15. *Математическая формулировка задачи*

Содержание углерода в процентах определяется по формуле $C = (0{,}8F_п + 6{,}67F_ц)/100$. Если $C > 2{,}14$, то это чугун, в противном случае – сталь.

Алгоритм решения задачи

1. Ввод исходных данных $F_п$ и $F_ц$. 2. Вычисление C. 3. Проверка условия $C > 2{,}14$; если оно выполняется, то вывод "чугун", в противном случае "сталь".

Схема программы, описывающей алгоритм, представлена на рис. 32.

16. *Математическая формулировка задачи*

Глубина закаленного слоя определяется по формуле

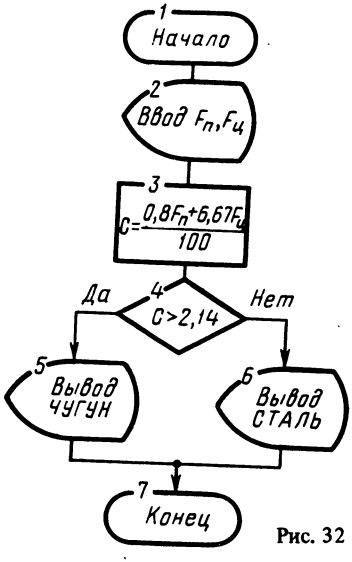


Рис. 32

$$Y = 5030\sqrt{\rho/(\mu f)},$$

где ρ — удельное электросопротивление; μ — абсолютная магнитная проницаемость; f — частота тока.

Алгоритм решения задачи

1. Ввод исходных данных ρ и μ. 2. Организация цикла по f от 50 до 400 Гц с шагом 10. 3. Вычисление Y. 4. Вывод значения Y. 5. Конец цикла.

Схема программы, описывающая алгоритм, представлена на рис. 33.

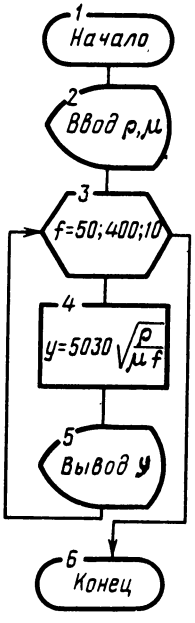


Рис. 33

17. *Математическая формулировка задачи*

Из уравнения фильтрации $t = (V^2 + 2cV)/k$. Изменяя значение переменной V от 0 до 20 с шагом 1, получить соответствующие значения времени фильтрации t.

Алгоритм решения задачи

1. Ввод исходных данных V_1, V_2, t_1, t_2. 2. Вычисление c, k. 3. Организация цикла, в котором изменяется параметр V от 0 до 20 с шагом 1. 4. Вычисление t. 5. Вывод значения результата t. 5. Конец цикла.

Схема программы, описывающая алгоритм, представлена на рис. 34.

18. *Математическая формулировка задачи*

Потери теплоты с 1 м^2 стенки определяются формулой $q = k(t_0 - t_3)$, где

$$k = \frac{1}{1/\alpha_1 + 1/\alpha_2 + h_1/\lambda_1 + h_2/\lambda_2}.$$

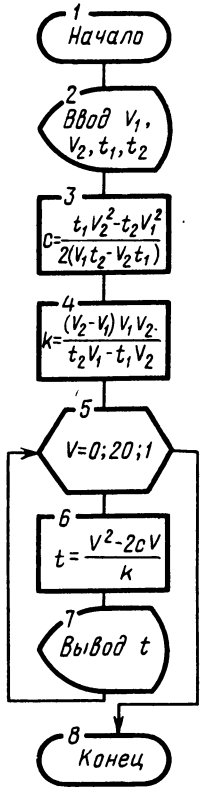


Рис. 34

Температура t_2 на границе между огнеупорным и строительным кирпичом находится из системы уравнений

$$\begin{cases} q = \alpha_1(t_0 - t_1); \\ q = \frac{\lambda_1}{h_1}(t_1 - t_2) \end{cases},$$

где $t_1 = t_0 - q/\alpha_1$ — температура внутренней стенки печи. Из системы уравнения получается $t_2 = t_1 - qh_1/\lambda_1$.

Строительный кирпич может использоваться при температуре, не превышающей 800° С.

Алгоритм решения задачи

1. Ввод исходных данных h_1, h_2, α_1, α_2, λ_1, λ_2, t_3. 2. Вычисление k. 3. Организация цикла с параметром t_0, изменяющимся от 100 до 3000°С с шагом 100° С. 4. Вычисление q, t_1, t_2. 5. Вывод q и t_2. 6. Проверка условия: если $t_2 < 800$°С, то повторять цикл, в противном случае выход из цикла. 7. Конец цикла. 8. Вывод t_0.

Схема программы, описывающая алгоритм, представлена на рис. 35. (вариант 1).

В предположении, что t_2 станет больше 800° С при $t_0 < 3000°$ С, выход из цикла возможен только по условию $t_2 > 800°$ С. В противном случае схема программы имеет вид, представленный на рис. 36 (вариант 2). В этом случае возможны два выхода из цикла: при выполнении условия $t_2 \geqslant 800°$ С, осуществляется вывод t_0 и естественный выход после выполнения цикла при $t_0 = 3000°$ С. В последнем случае условие $t_2 \geqslant 800°$ С не выполнено, поэтому вторая часть задачи не решена, о чем свидетельствует выводимое сообщение.

19. *Математическая формулировка задачи*

Скорость сушки определяется по формуле $v = \Delta W' / \Delta t$, где $\Delta W' = W'_{i-1} - W'_i$.

Критическое влагосодержание материала определяется по резкому уменьшению скорости сушки. Зависимость скорости сушки от времени представлена на рис. 37.

Поскольку значения v определяются по экспериментальным данным, то возможны отклонения отдельных точек графика от среднего значения. Поэтому для определения критической точки $t_{кр}$ используется относительное изменение скорости сушки

$$\frac{\Delta v}{v} = \frac{|v_i - v_{i-1}|}{v_i}.$$

Если относительное изменение скорости сушки больше по абсолютному значению 0,1, то критическая точка пройдена.

Алгоритм решения задачи

1. Ввод значений массива W, v_0. 2. Организация цикла по i от 1 до 20 с шагом 1. 3. Вычисление значения скорости сушки v, Δv. 4. Вывод v. 5. Проверка условия: если $\frac{\Delta v}{v} > 0{,}1$, то считать $W'_{кр} = W'_i$ и $t_{кр} = i$, в противном случае $W'_{кр}$ не вычислять. 6. Конец цикла. 7. Вывод $W'_{кр}$ и $t_{кр}$.

Схема программы, описывающая алгоритм, представлена на рис. 38.

Поскольку для решения задачи необходимо знать два значения скорости сушки – текущее v_i и предыдущее v_{i-1} (на схеме обозначено через v и v_0), пришлось в цикле сохранять это значение ($v_0 = v$) и ввести начальное значение v_0, равное значению $W'_0 - W_1$.

21. *Математическая формулировка задачи*

Количество глины, идущей на приготовление раствора, определяется из формулы

$$x = \frac{\gamma_1(\gamma_2 - \gamma)}{\gamma_1 - \gamma},$$

где γ – плотность воды; γ_2 – необходимая плотность раствора; γ_1 – плотность глины.

Алгоритм решения задачи

1. Ввод исходных данных γ, γ_1. 2. Организация цикла по параметру

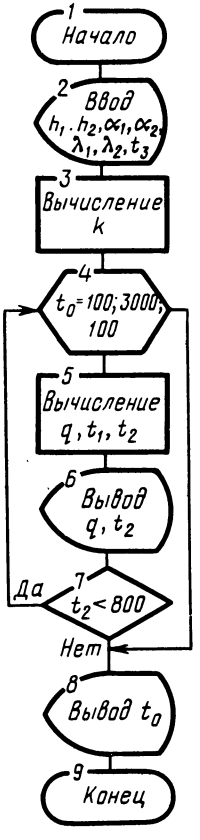


Рис. 35

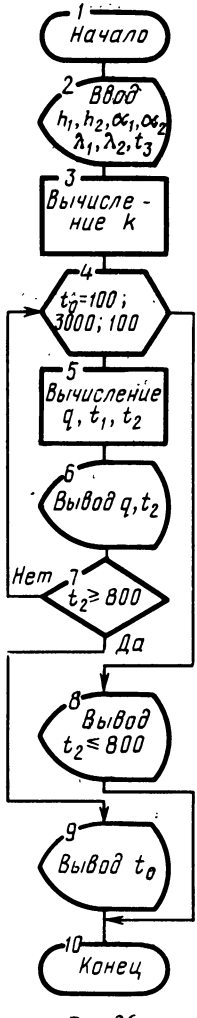


Рис. 36

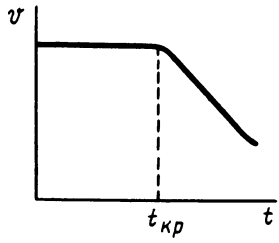


Рис. 37

γ_2, изменяющемуся от 1 до 2,5 с шагом 0,1. 3. Вычисление x. 4. Вывод x. 5. Конец цикла.

Схема программы, описывающая алгоритм, представлена на рис. 39.

22. *Математическая формулировка задачи*

Удельное сопротивление четырехосных вагонов массой q определяется по формуле

$$W_1 = 0{,}7 + \frac{8{,}0 + 0{,}1V + 0{,}0025V^2}{q/4}.$$

Удельное дополнительное сопротивление в зависимости от уклона и радиуса кривой равно: $i_0 = i + 700/R$.

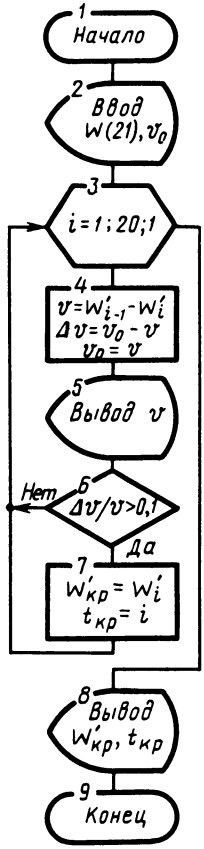


Рис. 38

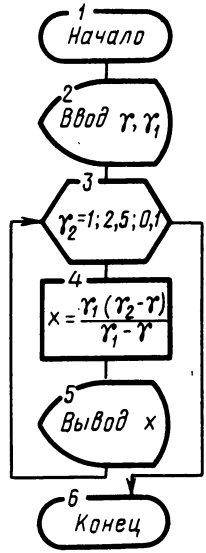


Рис. 39

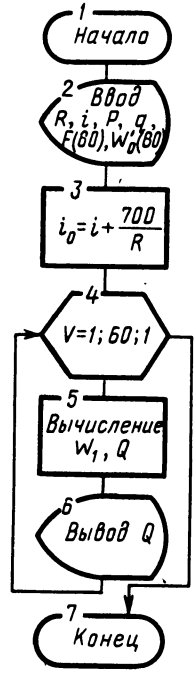


Рис. 40

Масса состава определяется следующим образом:

$$Q = \frac{F - P(W_0 + i_0)}{W_1 + i_0}$$

Алгоритм решения задачи

1. Ввод исходных данных R, i, P, q и значений элементов массивов F и W_0'. 2. Вычисление i_0. 3. Организация цикла по V от 1 до 60 с шагом 1. 4. Вычисление W_1, Q. 5. Вывод результатов Q. 6. Конец цикла.

Схема программы, описывающая алгоритм, представлена на рис. 40.

23. *Математическая формулировка задачи*

Расход электрической энергии определяется по формуле $A = \sum_{i=1}^{t} V_c I_m \Delta t/3600$, где I_m – ток, потребляемый из сети всеми тяговыми двигателями и устройствами питания собственных нужд в случае постоянного тока; V_c – напряжение питающей цепи; Δt – отрезок времени, для которого V_c и I_m условно принимаются постоянными.

Для расчета расхода электрической энергии потребляемый ток задается следующим образом:

i	1	2 ...	n	
I_m	I_{m_1}	I_{m_2}	I_{mn},	где $n=(t/\Delta t)+1$.

Алгоритм решения задачи

1. Ввод исходных данных t, Δt, V_c и значений элементов массива I_{mi}. 2. Задание начального значения суммы $S = 0$, вычисление n. 3. Организация цикла по i от 1 до n с шагом 1. 4. Накопление суммы. 5. Конец цикла. 6. Вычисление A. 7. Вывод результата A.

Схема программы, описывающая алгоритм, представлена на рис. 41.

24. *Математическая формулировка задачи*

Для решения задачи необходимо вычислить $d_{ср}=\frac{1}{10}\sum_{i=1}^{10} d_i$.

Алгоритм решения задачи

1. Ввод исходных данных P, D и значений элементов массива d. 2. Задание начального значения суммы $S = 0$. 3. Организация цикла по i от 1 до 10 с шагом 1. 4. Накопление суммы в цикле. 5. Конец цикла. 6. Вычисление $d_{ср}$ и НВ. 7. Вывод результата НВ.

Схема программы, описывающая алгоритм, представлена на рис. 42.

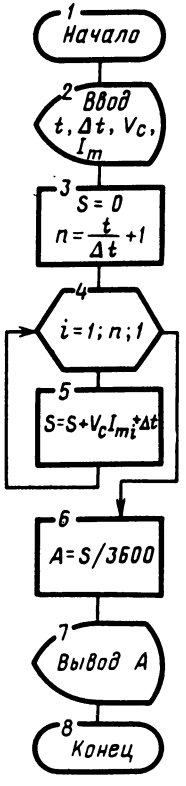


Рис. 41

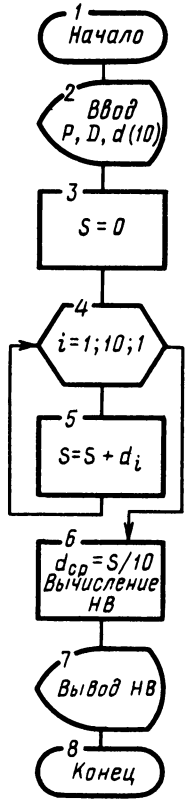


Рис. 42

25. *Математическая формулировка задачи*

$$\bar{T} = \frac{\sum_{i=1}^{n} m_i T_i}{\sum_{i=1}^{n} m_i}; \quad \bar{A} = \frac{\sum_{i=1}^{n} m_i A_i}{\sum_{i=1}^{n} m_i},$$

где $\bar{T}$ и $\bar{A}$ – средневзвешенные значения твердости и абразивности; T_i и A_i – текущие значения твердости и абразивности по результатам опыта; m_i – содержание в пробе пород твердостью T_i и абразивностью A_i.

Алгоритм решения задачи

1. Ввод исходных данных n и значений элементов массивов A_i, T_i, m_i. 2. Задание начальных значений сумм $\bar{T}$, $\bar{A}$ и $\bar{m}$, равных нулю. 3. Организация цикла по i от 1 до n с шагом 1. 4. Накопление сумм $\bar{T}$, $\bar{A}$, $\bar{m}$. 5. Конец цикла. 6. Вычисление $\bar{T}$ и $\bar{A}$. 7. Вывод результатов $\bar{T}$ и $\bar{A}$.

Схема программы, описывающая алгоритм, представлена на рис. 43.

26. *Математическая формулировка задачи*

$$\bar{X} = \frac{1}{n} \sum_{i=1}^{n} x_i; \quad S = \sqrt{\frac{\sum_{i=1}^{n} (\bar{X} - x_i)^2}{n-1}}.$$

Алгоритм решения задачи

1. Ввод исходных данных n и значений элементов массива X. 2. Задание начальных значений сумм $\bar{X}$ и S перед циклом, равных нулю. 3. Организация цикла по i от 1 до n с шагом 1. 4. Накопление в цикле суммы $\bar{X}$. 5. Конец цикла. 6. Вычисление среднего арифметического $\bar{X}$. 7. Организация цикла по i от 1 до n с шагом 1. 8. Накопление суммы S. 9. Конец цикла. 10. Вычисление S. 11. Вывод результатов $\bar{X}$ и S.

Схема программы, описывающая алгоритм, представлена на рис. 44.

27. *Математическая формулировка задачи*

Координаты центра тяжести определяются по формулам:

$$x_m = \frac{\sum_{i=1}^{n} x_i m_i}{\sum_{i=1}^{n} m_i}; \quad y_m = \frac{\sum_{i=1}^{n} y_i m_i}{\sum_{i=1}^{n} m_i}; \quad z_m = \frac{\sum_{i=1}^{n} z_i m_i}{\sum_{i=1}^{n} m_i}.$$

Алгоритм решения задачи

1. Ввод исходных данных n и массивов X, Y, Z, M. 2. Задание начального значения сумм x_m, y_m, z_m, $\bar{m}$, равных нулю. 3. Организация цикла по i от 1 до n с шагом 1. 4. Вычисление сумм x_m, y_m, z_m, $\bar{m}$. 5. Конец цикла. 6. Вычисление координат x_m, y_m, z_m. 7. Вывод результатов x_m, y_m, z_m.

Схема программы, описывающая алгоритм, представлена на рис. 45.

28. *Математическая формулировка задачи*

При вычислении требуемой поверхности воспользуемся методом прямоугольников, согласно которому

$$v = \Delta T \sum_{i=1}^{n} \frac{1}{k_i (T_i - t_i)}, \text{ где } n = \left[\frac{T_1 - T_2}{\Delta T}\right] + 1.$$

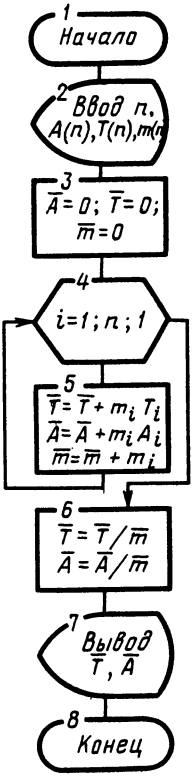


Рис. 43

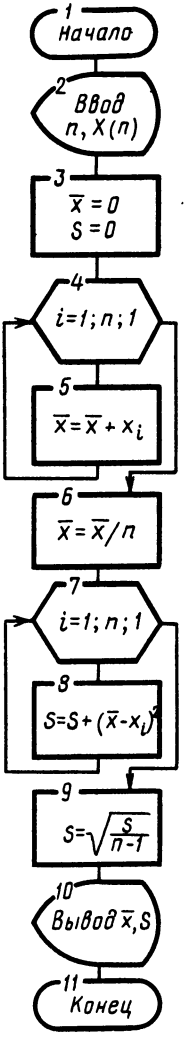


Рис. 44

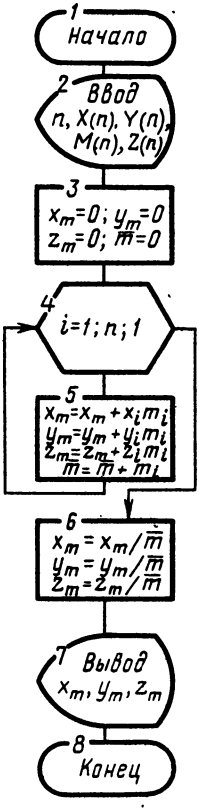


Рис. 45

Для вычисления суммы необходимо найти значения t_i. Из уравнения теплового баланса $Gc(T_1 - T_i) = G_x c_x (t_2 - t_i)$ найти, подставляя вместо T_i и t_i крайние значения температур горячего и холодного масла, значение коэффициента $A = \frac{Gc}{G_x c_x} = \frac{t_2 - t_1}{T_1 - T_2}$. Следовательно, $t_i = t_2 - A(T_2 - T_i)$.

Алгоритм решения задачи

1. Ввод исходных данных G, c, t_2, T_1, T_2, t_1 и табличные значения переменной k_i. 2. Вычисление значения коэффициента A, $y = 0$ и $j = 0$. 3. Организация цикла, в котором изменяется параметр T от T_1 до T_2 с

шагом $\Delta T = -10$. 4. В цикле вычисление j и выбор значения k_j, соответствующего T_j, вычисление t_j, накопление суммы y. 5. Конец цикла. 6. Вычисление $F = 10Gcy$. 7. Вывод результата F.

Схема программы, описывающей алгоритм, представлена на рис. 46.

29. *Математическая формулировка задачи*

$$1/R_0 = 1/R_1 + 1/R_2 + \ldots + 1/R_n = \sum_{i=1}^{n} 1/R_i.$$

Для вычисления общего сопротивления цепи удобно использовать прием накопления суммы.

Алгоритм решения задачи

1. Ввод исходных данных n и значений элементов массива R_i. 2. Задание начального значения S, равное нулю. 3. Организация цикла по i от 1 до n с шагом 1. 4. Вычисление суммы S. 5. Конец цикла. 6. Вычисление R_0. 7. Вывод результата R_0.

Схема программы, описывающая алгоритм, представлена на рис. 47.

30. *Математическая формулировка задачи*

Решение задачи сводится к накоплению суммы и произведения.

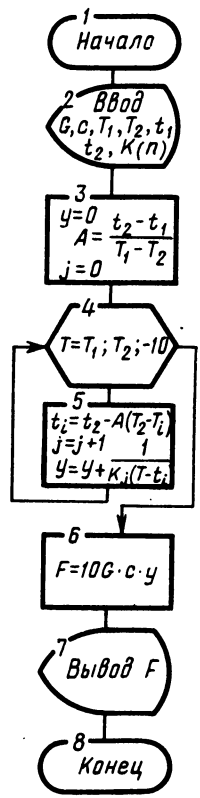


Рис. 46

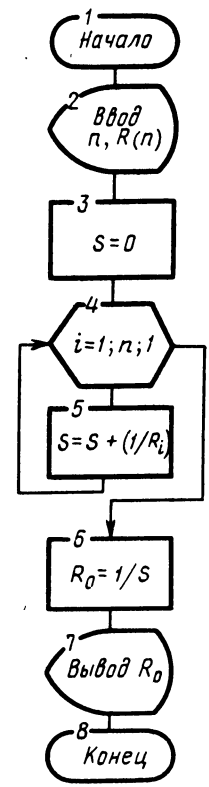


Рис. 47

Алгоритм решения задачи

1. Ввод исходных данных n и значений элементов массива R_i. 2. Задание начального значения $P = 1$. 3. Организация цикла по i от 1 до n с шагом 1. 4. Накопление произведения P. 5. Конец цикла. 6. Задание начального значения $S = 0$. 7. Организация цикла по j от 1 до n с шагом 1. 8. Накопление суммы S. 9. Конец цикла по j. 10. Вычисление R_0. 11. Вывод результата R_0.

Схема программы, описывающая алгоритм, представлена на рис. 48.

31. *Математическая формулировка задачи*

Решение этой задачи требует вычисления $N!$. Вычисление факториала сведем к вычислению произведения $M = \prod_{i=1}^{N} i$.

Алгоритм решения задачи

1. Ввод исходных данных N. 2. Задание начального значения произведения $M = 1$. 3. Организация цикла по i от 1 до N с шагом 1. 4. Накопление произведения. 5. Конец цикла. 6. Вывод результата M.

Схема программы, описывающей алгоритм, представлена на рис. 49.

32. *Математическая формулировка задачи*

Определение количества способов сводится к вычислению числа сочетаний из n элементов по m по формуле $C_n^m = \frac{n!}{m!(n-m)!}$. Подставляя вместо факториалов их значения, получим $C_n^m = \prod_{i=1}^{n-m} \frac{m+i}{i}$, т. е. решение задачи сводится к накоплению произведения.

Алгоритм решения задачи

1. Ввод исходных данных n и m. 2. Задание начального значения произведения $P = 1$. 3. Организация цикла по i от 1 до $n - m$ с шагом 1. 4. Накопление произведения. 5. Конец цикла. 6. Вывод P.

Схема программы, описывающая алгоритм, представлена на рис. 50.

33. *Математическая формулировка задачи*

Значения H_i и N_i в таблице испытаний определены как функция Q_i, изменяющегося от 0 до 50 с шагом 5. Коэффициент полезного действия насоса определяется формулой $\eta = QH\gamma/(102N)$. Вычисляя η_i для различных значений Q_i, H_i, N_i, получим зависимость η_i от Q_i. Для нахождения наибольшего значения η воспользуемся формулой

$$\eta_{\max} = \begin{cases} \eta_i, \text{ если } \eta_i > \eta_{\max}, \\ \eta_{\max}, \text{ если } \eta_i \leqslant \eta_{\max}, \end{cases}$$

которая имеет следующий смысл: если вновь вычисленное значение η_i больше максимального из всех предыдущих значений, то считать $\eta_{\max}$ равным этому η_i, в противном случае $\eta_{\max}$ сохраняет свое прежнее значение.

Алгоритм решения задачи

1. Ввод исходных данных γ и массивов H и N. 2. Задание начальных значений $i = 0$ и $\eta_{\max} = -10^{10}$. Организация цикла при изменении Q от 0 до 50 с шагом 5. 4. Вычисление η_i и i. 5. Проверка условия $\eta_i > \eta_{\max}$; если оно выполняется, то вычисляются $Q_{\max}$, i_m, и цикл повторяется;

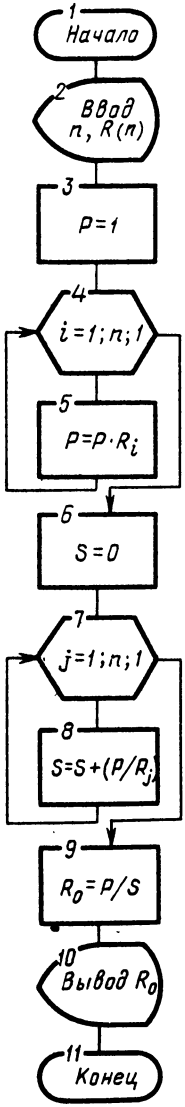


Рис. 48

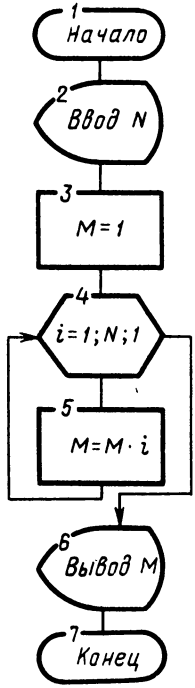


Рис. 49

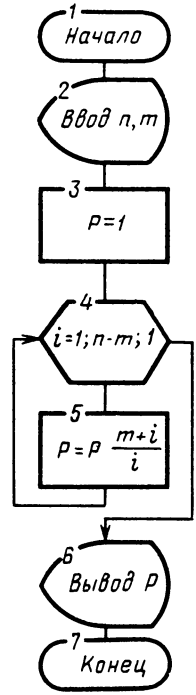


Рис. 50

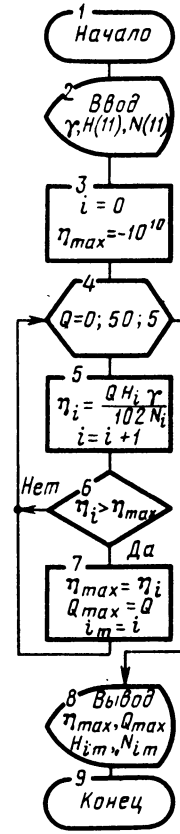


Рис. 51

в противном случае производится переход к началу цикла. 6. Конец цикла. 7. Вывод $\eta_{\max}$, N_{i_m}, H_{i_m}, $Q_{\max}$.

Схема программы, описывающая алгоритм, представлена на рис. 51.

34. *Математическая формулировка задачи*

Количество воздуха определяется по:

1) газовыделению (метана или углекислоты)

$$Q_1 = q_m TZ;$$

2) расходу взрывчатых веществ

$$Q_2 = \frac{12{,}5\,A\,b\,Z}{t};$$

3) наибольшему числу людей, находящихся одновременно в шахте,

$$Q_3 = qmZ.$$

В этих формулах q — норма воздуха на одного работающго под землей; m — наибольшее количество людей, работающих в смену; Z — коэффициент запаса воздуха; b — количество условной окиси углерода, образовавшейся при взрывании 1 кг взрывчатых веществ (принимается согласно правилам техники безопасности равным 40 л на 1 кг); t — время проветривания забоев; q_m — норма подачи воздуха на 1 т суточной добычи шахты; T — суточная добыча шахты; A — расход взрывчатых веществ за один цикл взрывания.

Для дальнейших расчетов принимают наибольшее значение из Q_1, Q_2 и Q_3. Проверка концентрации метана в исходящей струе из участка выполняется по формуле

$$h_1 = \frac{q_{CH_4}\, T_{уч}}{14{,}4\, q_{уч}},$$

где q_{CH_4} — относительная газообильность шахты по метану (12 м^3/т); $T_{уч}$ — суточная добыча участка; $q_{уч}$ — количество воздуха, потребное для проветривания участка.

Правилами техники безопасности допускается содержание метана в исходящей струе из участка не более 1%.

Депрессия выработок определяется по формуле

$$h_2 = \frac{a_1 P L}{S^3} Q^2,$$

где a_1 — коэффициент аэродинамического сопротивления, характеризующий шероховатость стенок выработки; P — периметр выработки; L — длина выработки; S — площадь поперечного сечения; Q — количество воздуха, поступающего в данную выработку.

Алгоритм решения задачи

1. Ввод исходных данных q, q_m, T, Z, A, b, t, m, q_{CH_4}, a_1, P, L, $T_{уч}$, S, $q_{уч}$. 2. Вычисление Q_1, Q_2, Q_3 и запись их в массив Q. 4. Задание начального значения $Q_{max} = Q_1$. 5. Организация цикла по i от 2 до 3 с шагом 1. 6. Проверка условия $Q_i > Q_{max}$, если оно выполняется, то вычисляется $Q_{max} = Q_i$, в противном случае — переход к началу цикла. 7. Конец цикла. 8. Вычисление h_1, h_2. 9. Вывод результатов Q_{max}, h_1, h_2.

Схема программы, описывающая алгоритм, представлена на рис. 52.

35. *Математическая формулировка задачи*

Метацентрический радиус определяется из формулы $r = I_x/V$, где I_x — момент инерции площади ватерлинии; V — объемное водоизмещение судна.

$$I_x = \frac{2}{3} \cdot \Delta L \left[\sum_{i=0}^{n} y_i^3 - \frac{1}{2} (y_0^3 - y_n^3) \right].$$

Алгоритм решения задачи

1. Ввод исходных данных n, ΔL, V, массива Y. 2. Задание начального значения суммы $S = 0$. 3. Организация цикла по i от 1 до n. 4. Накопление суммы S. 5. Конец цикла. 6. Вычисление I_x и r. 7. Вывод результата r.

Схема программы, описывающая алгоритм, представлена на рис. 53.

36. *Математическая формулировка задачи*

Зависимость давления P от объема V цилиндра, снятая экспериментально, сведена в таблицу при изменении аргумента V с шагом ΔV. Если по результатам эксперимента построить график зависимости P от V, то он имеет вид, представленный на рис. 54. Стрелки на графике обозначают направление изменения V. Мощность цилиндра S определяется

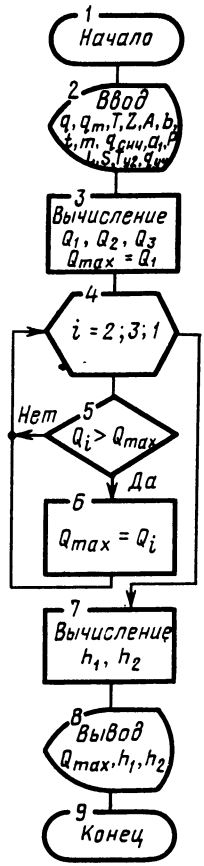


Рис. 52

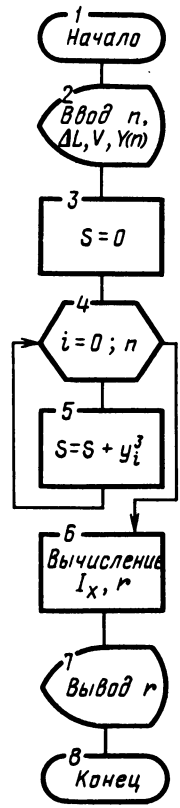


Рис. 53

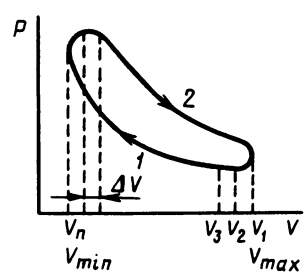


Рис. 54

площадью, заключенной внутри кривой. Следовательно, решение задачи сводится к отысканию площади. Если площадь под кривой *1* обозначить S_1, а площадь под кривой *2* обозначить S_2, тогда искомая площадь равна их разности $S = S_2 - S_1$. Для вычисления площади используют способ прямоугольников, при котором площадь под кривой представляется суммой площадей элементарных прямоугольников шириной ΔV. Тогда $S \approx \Delta V \sum_{i=1}^{n} P_{n+i} - \Delta V \sum_{i=1}^{n} P_i$, где n – количество элементарных прямоугольников,

$$n = \frac{V_{max} - V_{min}}{\Delta V} + 1 = \frac{V_1 - V_n}{\Delta V} + 1.$$

В таблице значения P_i расположены в последовательности, указанной стрелками на рисунке, и составляют $2n$ значений при изменении аргумента V от V_{max} до V_{min} и обратно. Таким образом, задача состоит в том, чтобы отыскать значения двух сумм S_1 и S_2, а затем вычислить S.

Алгоритм решения задачи

1. Ввод исходных данных n, ΔV и значений элементов массива P, состоящего из $2n$ элементов. 2. Задание начальных значений сумм S_1 и S_2, равных нулю. 3. Организация цикла по i от 1 до n с шагом 1. 4. Накопление суммы S_1. 5. Конец первого цикла. 6. Организация второго цикла по i от $n + 1$ до $2n$ с шагом 1. 7. Накопление суммы S_2. 8. Конец второго цикла. 9. Вычисление S. 10. Вывод результата S.

Решение задачи потребует меньше времени, если обе суммы вычислять в одном цикле по i от 1 до n, тогда в цикле вычисляются суммы $S_1 = S_1 + P_i$ и $S_2 = S_2 + P_{n+i}$.

Схема программы, описывающая алгоритм для последнего случая, представлена на рис. 55.

37. *Математическая формулировка задачи*

Зависимость абсолютной скорости от составляющих определяется формулой

$$V_и = \sqrt{V_0^2 + V_т^2 + 2V_0V_т \cos q_т}.$$

Изменяя значение $q_т$ от 0 до 2π с шагом $\Delta q_т$, получить зависимость $V_и$ от $q_т$. Изменяя V_0 на ΔV_0, получить новую зависимость.

Алгоритм решения задачи

1. Ввод исходных данных $\Delta q_т$, $V_т$, ΔV_0, $V_{0\,min}$, $V_{0\,max}$. 2. Организация цикла, в котором изменяется V_0 от $V_{0\,min}$ до $V_{0\,max}$ с шагом ΔV_0. 3. Организация цикла, в котором изменяется $q_т$ от 0 до 2π с шагом $\Delta q_т$. 4. Вычисление $V_и$. 5. Вывод $V_и$. 6. Конец цикла по $q_т$. 7. Конец цикла по V_0.

Схема программы, описывающая алгоритм, представлена на рис. 56.

38. *Математическая формулировка задачи*

Эмпирическая формула для определения просадки кормы имеет вид

$$\Delta T_k = \begin{cases} 0{,}00135\,k\,V^2/g\,[16{,}43 - (L/B)]\sqrt{T/H}, & \text{если } H/T \leqslant 1{,}6, \\ 0{,}00112\,k\,V^2/g\,[16{,}43 - (L/B)]\sqrt{T/H}, & \text{если } H/T > 1{,}6, \end{cases}$$

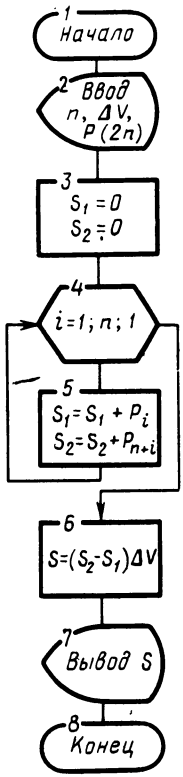


Рис. 55

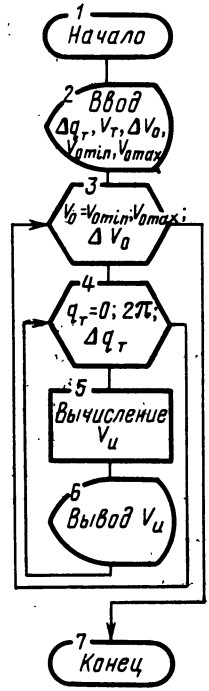


Рис. 56

где g — ускорение свободного падения; k — числовой коэффициент, принимающий для винтовых судов следующие значения:

$$k = \begin{cases} 1{,}15, \text{ если } (L/B) < 7, \\ 1{,}1, \text{ если } (L/B) \geqslant 7. \end{cases}$$

Полагаем, что L/B не может быть меньше 5 и больше 9.

Алгоритм решения задачи

1. Ввод исходных данных g, T, H, L, B, V_{min}, V_{max}, ΔV. 2. Проверка условия $(L/B) < 7$; если оно выполняется, то задать $k = 1{,}15$, в противном случае, $k = 1{,}1$. 3. Организация цикла по V от V_{min} до V_{max} с шагом ΔV. 4. Проверка условия $(H/T) \leqslant 1{,}6$; если оно выполняется, то вычислить $\Delta T_k = 0{,}00135\, k V^2/g\,[16{,}43 - (L/B)]\sqrt{T/H}$, в противном случае $\Delta T_k = 0{,}00112\, k V^2/g\,[16{,}43 - (L/B)]\sqrt{T/H}$. 5. Вывод значения ΔT_k. 6. Конец цикла.

Схема программы, описывающая алгоритм, представлена на рис. 57.

39. *Математическая формулировка задачи*

Значения сил веса, соответствующие эпюре весовой нагрузки, сводятся в массив G, а сил поддержания – в массив P, где $P_i = \gamma V_i$, γ – плотность воды, V_i – объем вытесненной жидкости. Решение задачи сводится к нахождению центра приложения X_G и X_P равнодействующих сил по формулам:

$$X_G = \frac{\bar{X}_G}{\bar{G}}\ ;\ X_P = \frac{\bar{X}_P}{\bar{P}}\ ,\ \text{где}\ \bar{X}_G = \sum_{i=1}^{n} G_i x_i\ ,$$
$$\bar{X}_P = \sum_{i=1}^{n} P_i x_i\ ,\ \bar{G} = \sum_{i=1}^{n} G_i,\ \bar{P} = \sum_{i=1}^{n} P_i\ ,$$

x_i – координаты приложения сил.

Алгоритм решения задачи

1. Ввод исходных данных n массивов G, P, X. 2. Задание начальных значений сумм $\bar{X}_G$, $\bar{X}_P$, $\bar{G}$, $\bar{P}$, равных нулю. 3. Организация цикла по i от 1 до n с шагом 1. 4. Вычисление $\bar{G}, \bar{P}, \bar{X}_G, \bar{X}_P$. 5. Конец цикла. 6. Вычисление X_G и X_P. 7. Вывод результатов X_G, X_P.

Схема программы, описывающая алгоритм, представлена на рис. 58.

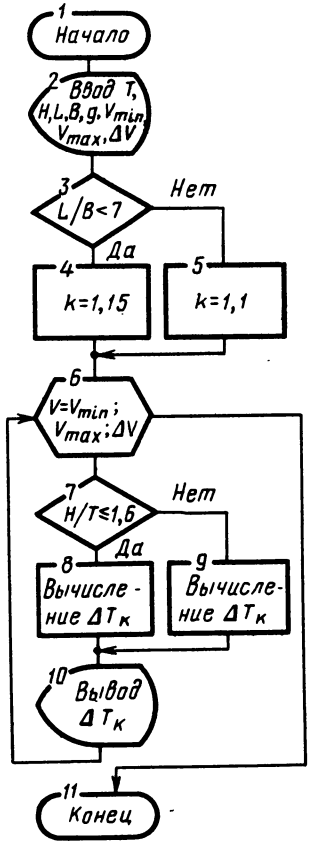


Рис. 57

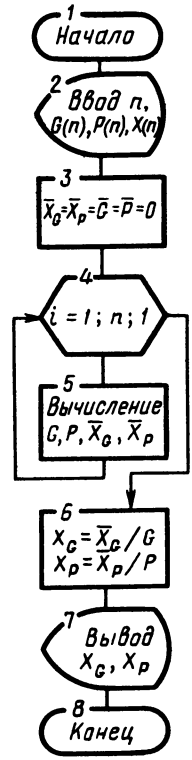


Рис. 58

53.

```
05 REM СКОРОСТЬ РАСТВОРА
10 READ D1,D2,Q
20 DATA ...
30 P=3.14159
40 F1=P*D1↑2/4
50 F2=P*D2↑2/4
60 F=F1-F2
70 V=Q/F
80 PRINT 'V=';V
90 END
```

В программе значение π присвоено переменной P, что позволило несколько сократить длину операторов, вычисляющих $F1$ и $F2$. Все остальные операторы записаны в соответствии со схемой программы (рис. 19).

54.

```
05 REM ДИАМЕТРЫ ЗУБЧАТЫХ КОЛЕС
10 READ N,M,A
20 DATA ...
30 Z1=2*A/(M*(N+1))
40 Z2=2*A*N/(M*(N+1))
50 D1=M*(Z1+2)
60 D2=M*(Z2+2)
70 D3=M*(Z1-2.5)
80 D4=M*(Z2-2.5)
90 PRINT D1,D2,D3,D4
100 END
```

55.

```
10 REM ПОВЕРХНОСТЬ ФИЛЬТРА
20 READ V0,X,P,P1,T,K1,C,H0,H1,X1
30 DATA ...
40 K=K1*P/P1
50 V=C+SQR(C*C+K*T)
60 M=1/(1-X1/100)
70 G=V0*H0
80 G0=G*X*M/100
90 G1=G-G0
100 F1=G1*T/(3600*H1*V)
110 F=F1/0.35
120 N=60*0.35/T
130 PRINT 'F=';F;'N=';N
140 END
```

63.

```
10 REM КРУТИЗНА ПУТИ
20 READ N1,Q,K,P,F1
30 DATA ...
40 Q0=Q/K
50 IF N1=0 THEN 80
60 W1=28/(Q0+7)
70 GOTO 90
80 W1=142/(Q0+7)
90 I=F1/(Q+P)-W1
100 PRINT 'I=';I
110 END
```

Здесь условный оператор с меткой 50 организует разветвление. В одной ветви вычисляется W1 для случая $n_1 = 1$, а во второй – для $n_1 = 0$. После любой ветви вычисляется значение I.

64.

```
10 REM СОПРОТИВЛЕНИЕ ДВИЖЕНИЮ
20 READ P,Q,N,V,R,S,L,I
30 DATA ...
40 W1=(1.9+0.01*V+0.0003*V*V)*P
50 QO=Q/(4*N)
60 W2=0.7+(8+0.1*V+0.0025*V*V)/QO
70 IF L<=S THEN 100
80 WO=700*S/R/L
90 GOTO 110
100 WO=700/R
110 W3=(P+Q)*(I+WO)
120 W=W1+W2+W3
130 PRINT 'W=';W
140 END
```

Условный оператор TF организует разветвление по условию $L \leqslant S$. Если оно выполняется, то осуществляется переход к оператору с меткой 100, в противном случае выполняется оператор, следующий за условным, а затем оператор GOTO, осуществляющий переход к оператору с меткой 110.

65.

```
10 REM КОЭФФИЦИЕНТ ТЕПЛООТДАЧИ
20 DIM Y(77)
30 READ D,W,TO,T3,C1,MO,G,P,PO,E,L,A,B,C
40 DATA ...
50 FOR I=1 TO 77
60 READ Y(I)
70 NEXT I
80 R=W*D*C1/MO/G
90 IF R>10000 THEN 140
100 IF R<2300 THEN 160
110 I=INT((R-2300)/100)
120 N=Y(I)*P↑0.43*(P/PO)↑0.25
130 GOTO 200
140 N=0.021*E*R↑0.8*P↑0.43*(P/PO)↑0.25
150 GOTO 200
160 T1=TO+1
170 T2=TO-1
180 B1=(A*(T1-T2)+B*(T1↑2-T2↑2)+C*(T1↑3-T2↑3))/(T1-T2)
190 N=0.15*E*R↑0.33*P↑0.43*G1↑0.1*(P/PO)↑0.25
200 A1=N*L/D
210 PRINT 'A1=';A1;'RE=';R
220 END
```

Для ввода значений массива Y организован цикл, в котором изменяется параметр I от 0 до 77. В операторе DATA располагаются значения переменных, указанных в первом операторе READ, а затем элементы массива Y.

В программе имеется три ветви. Первая ветвь начинается оператором с меткой 110 и в операторе с меткой 120 вычисляется значение N для случая $2300 \leqslant R \leqslant 10\,000$ (переходный режим). Вторая ветвь начинается оператором с меткой 140, в котором вычисляется значение N для случая $R > 10\,000$ (турбулентный режим). В третьей ветви, начинающейся оператором с меткой 160, вычисляется N для случая $R < 2300$ (ламинарный режим). После выполнения операторов любой ветви организуется переход к оператору с меткой 200.

108.

```
10 REM ФИЛЬТРАЦИЯ
20 READ V1,V2,T1,T2
30 DATA ...
40 C=(T1*V2↑2-T2*V1↑2)/2/(V1*T2-V2*T1)
50 K=((V1-V2)*V1*V2)/(T2*V1-T1*V2)
60 FOR V=0 TO 20
70 T=(V*V+2*V*C)/K
80 PRINT 'V=';V;'T=';T
90 NEXT V
100 END
```

109. Программа для первого варианта (см. рис. 35) имеет вид

```
10 REM ПОТЕРИ ТЕПЛА
20 READ H1,H2,A1,A2,L1,L2,T3
30 DATA ...
40 K=1/(1/A1+1/A2+H1/L1+H2/L2)
50 FOR T0=100 TO 3000 STEP 100
60 Q=K/(T0-T3)
70 T1=T0-Q/A1
80 T2=T1-Q*H1/L1
90 PRINT 'Q=';Q;'T2=';T2
100 IF T2>=800 THEN 120
110 NEXT T0
120 PRINT 'T0=';T0
130 END
```

Для второго варианта (см. рис. 36) часть программы, начиная с оператора с меткой 100, имеет вид

```
100 IF T2>=800 THEN 140
110 NEXT T0
120 PRINT 'T2<800'
130 GOTO 150
140 PRINT 'T0=';T0
150 END
```

111.

```
10 REM СКОРОСТЬ СУШКИ
20 DIM W(20)
30 FOR I=0 TO 20
40 READ W(I)
50 DATA ...
60 NEXT I
70 V0=W(0)
80 FOR I=1 TO 20
90 V=W(I-1)-W(I)
100 PRINT 'V=';V
110 K=(V0-V)/V
120 IF K<=0.1 THEN 150
130 W0=W(I)
140 T0=I
150 NEXT I
160 PRINT 'WKR=';W0;'TKR=';T0
170 END
```

112.

```
10 REM РАСХОД ЭНЕРГИИ
20 DIM J(100)
30 READ T,H,V
40 DATA ...'
50 N=T/H+1
60 S=0
70 FOR I=1 TO N
80 READ J(I)
90 S=S+V*H*J(I)
100 NEXT I
110 A=S/3600
120 PRINT 'A=';A
130 END
```

Таблица состоит из $n + 1$ элемента. В операторе DIM размер массива можно задавать только числом, поэтому, записав в этом операторе 100, введено ограничение на величину n, которое не может быть больше 100. Элемент массива, имеющий индекс 0, в программе не используется.

113.

```
10 REM ТВЕРДОСТЬ И АБРАЗИВНОСТЬ
20 DIM M(100),A(100),T(100)
30 READ N
40 DATA ...
50 MO=0
60 AO=0
70 TO=0
80 FOR I=1 TO N
90 READ M(I),A(I),T(I)
100 MO=MO+M(I)
110 AO=AO+A(I)*M(I)
120 TO=TO+T(I)*M(I)
130 NEXT I
140 AO=AO/MO
150 TO=TO/MO
160 PRINT AO,TO
170 END
```

В операторе DATA сначала должно быть указано значение N, затем элементы M(1), A(1), T(1), M(2), A(2), T(2), ... Элементы массивов с индексом 0 не используются.

114.

```
10 REM СРЕДНИЕ ОТКЛОНЕНИЯ
20 DIM X(100)
30 READ N
40 DATA ...
50 XO=0
60 S=0
70 FOR I=1 TO N
80 READ X(I)
90 XO=XO+X(I)
100 NEXT I
110 XO=XO/N
120 FOR I=1 TO N
130 S=S+(XO-X(I))↑2
140 NEXT I
150 S=SQR(S/(N-1))
160 PRINT 'XO=';XO;'S=';S
170 END
```

Здесь используется два независимых цикла, поэтому можно использовать для обоих циклов один и тот же параметр I. В операторе DATA сначала указывается значение N, а затем элементы массива X. Элемент массива с индексом 0 не используется.

115.

```
10 REM ЦЕНТР ТЯЖЕСТИ
20 DIM M(100),X(100),Y(100),Z(100)
30 READ N
40 DATA ...
50 MO=0
60 XO=0
70 YO=0
80 ZO=0
90 FOR I=1 TO N
100 READ M(I),X(I),Y(I),Z(I)
110 MO=MO+M(I)
120 XO=XO+X(I)*M(I)
130 YO=YO+Y(I)*M(I)
140 ZO=ZO+Z(I)*M(I)
150 NEXT I
160 XO=XO/MO
170 YO=YO/MO
180 ZO=ZO/MO
190 PRINT 'XM=';X;'YM=';YO;'ZM=';ZO
200 END
```

В операторе DATA сначала указывается значение переменной N, а затем элементы массивов M(1), X(1), Y(1), Z(1), M(2), X(2), Y(2), Z(2), . . . , M(N), X(N), Y(N), Z(N).

116.

```
10 REM ТЕПЛООБМЕННИК
20 DIM K(20)
30 READ G,C,T1,T2,X1,X2
40 DATA ...
50 Y=0
60 A=(X2-X1)/(T1-T2)
70 V=0
80 FOR T=T1 TO T2 STEP -10.
90 J=J+1
100 READ K(J)
110 X=X2-A*(T2-T)
120 Y=Y+1/(K(J)*(T-X))
130 NEXT T
140 F=G*C*Y*10
150 PRINT 'F=';F
160 END
```

117.

```
10 REM ЦЕНТРОБЕЖНЫЙ НАСОС
20 DIM H(11),N(11)
30 READ X
40 DATA ...
50 FOR I=1 TO 11
60 READ H(I),N(I)
70 NEXT I
80 I=1
90 YO=-1E10
100 FOR Q=0 TO 50 STEP 5
110 Y=Q*H(I)*X/(102*N(I))
```

```
120 IF Y YO THEN 160
130 YO=Y
140 IO=I
150 QO=Q
160 I=I+1
170 NEXT Q
180 PRINT YO;QO;N(IO);H(IO)
190 END
```

Здесь в цикле изменяются два параметра: переменная Q от 0 до 50 с шагом 5; переменная I – от 1 до 11 с шагом 1. В операторе цикла задан закон изменения Q, параметр I изменяется в цикле с помощью оператора присваивания. В цикле вычисляется Y, находится его наибольшее значение YO и значения параметров QO и IO, при которых оно получено.

118.

```
10 REM MAXTA
20 DIM Q(3)
30 READ QO,Q1,T,Z,A,B,TO,M,Q2,A1,P,L,S,T4,Q4
40 DATA ...
50 Q(1)=Q1*T*Z
60 Q(2)=12.55*A*B*Z/TO
70 Q(3)=QO*M*Z
80 Q3=Q(1)
90 FOR I=2 TO 3
100 IF Q(I)<=Q3 THEN 120
110 Q3=Q(I)
120 NEXT I
130 H1=Q2*T4/(14.4*Q4)
140 H2=A1*P*L*Q3↑2/S↑3
150 PRINT 'Q MAX=';Q3;'H1=';H1;'H2=';H2
160 END
```

В цикле с помощью условного оператора находится наибольшее значение Q. Для этого перед циклом задается наибольшее значение наибольшего Q3 = Q(1). В цикле при невыполнении условия Q(I) < Q3 имя Q3 получает значение Q(I), так как оно больше предыдущего его значения.

119.

```
10 REM МЕТАЦЕНТРИЧЕСКИЙ РАДИУС
20 DIM Y(100)
30 READ N,L,V
40 DATA...
50 S=0
60 FOR I=1 TO N
70 READ Y(I)
80 S=S+Y(I)↑3
90 NEXT I
100 IO=2/3*L*(S-(Y(0)↑3+Y(N)↑3)/2)
110 R=IO/V
120 PRINT 'R=';R
130 END
```

120.

```
10 REM МОЩНОСТЬ ДВИГАТЕЛЯ
20 DIM P(100)
30 READ N,VO
```

```
40 DATA ...
50 N=2*N
60 FOR I=1 TO N
70 READ P(I)
80 NEXT I
90 S1=0
100 S2=0
110 FOR I=1 TO N
120 S1=S1+P(I)
130 S2=S2+P(N+I)
140 NEXT I
150 S=(S2-S1)*V0
160 PRINT 'S=';S
170 END
```

В программе два цикла. Первый цикл используется для ввода элементов массива P, состоящего из $2n$ элементов, которые размещаются в памяти машины в ячейках с первой по $2n$. Второй цикл используется для накопления сумм S1 и S2, каждая сумма состоит из n элементов, поэтому S1 – сумма элементов, начиная с первого по n, а S2 – сумма элементов с $n+1$ по $2n$. При составлении программы полагалось, что n известно. Если известно количество значений в диаграмме, то получить n несложно, разделив это число пополам. Если же известно V_{max} и V_{min}, то целесообразно включить в программу оператор присваивания, вычисляющий n по приведенной в ответе к задаче 36 формуле. В операторе DATA сначала должно быть указано значение N, а затем значения $2n$ элементов массива P.

121.

```
10 REM ПРОСАДКА СУДНА
20 READ G,T,H,L,B,V0,V1,V2
30 DATA...
40 IF L/B<7 THEN 70
50 K=1.1
60 GOTO 80
70 K=1.15
80 FOR V=V0 TO V1 STEP V2
90 IF H/T>1.6 THEN 120
100 T2=0.00135*K*V*V/G*(16.43-L/B)*SQR(T/H)
110 GOTO 130
120 T2=0.00112*K*V*V/G*(16.43-L/B)*SQR(T/H)
130 PRINT 'T2=';T2
140 NEXT V
150 END
```

Программа составлена в соответствии со схемой (см. рис. 57). Однако ее объем можно несколько сократить. Так, вместо операторов с метками 40, 50, 60, 70 можно записать такую последовательность операторов:

```
40 K=1.15
50 IF L/B<7 THEN 80
60 K=1.1
80 FOR ...
```

Здесь сначала K присваивается значение 1.15. Если $L/B < 7$, то осуществляется переход к метке 80 (при этом K сохраняет свое значение). Если же условие $L/B < 7$ не выполняется, то осуществляется переход к оператору с меткой 60, присваивающему K новое значение 1.1.

В операторах с метками 100 и 120 вычисляется одно и то же выражение, отличающееся лишь коэффициентом, поэтому целесообразно перед условным оператором с меткой 90 вычислить `90 T3=K*V*V/G*(16.43-L/B)*SQR(T/H)`, а затем при вычислении

Т2 исполь зовать значение Т3, т. е. операторы, вычисляющие Т2, будут иметь вид

```
100 T2=0.00135*T3
120 T2=0.00112*T3
```

122.

```
10 REM СУДНО
20 DIM G(100),P(100),X(100)
30 READ N
40 DATA ...
50 XO=O
60 X1=O
70 GO=O
80 PO=O
90 FOR I=1 TO N
100 READ G(I),P(I),X(I)
110 XO=XO+G(I)*X(I)
120 X1=X1+P(I)*X(I)
130 GO=GO+G(I)
140 PO=PO+P(I)
150 NEXT I
160 XO=XO/GO
170 X1=X1/PO
180 PRINT XO,X1
190 END
```

В операторе DATA сначала должно быть записано значение N, а затем значения элементов массивов в следующем порядке: G(1), P(1), X(1), G(2), P(2), X(2), ..., G(100), P(100), X(100).

При этом нулевые элементы массивов использоваться не будут.

ПРИЛОЖЕНИЕ 1

Работа с микроЭВМ.
ЭВМ типа ДВК. Средства редактирования программ

Для выполнения вспомогательных операций, связанных с редактированием текста программы, пуском и остановом выполнения, служат команды языка БЕЙСИК и специальные клавиши на пульте ЭВМ.

1. **Стирание символа** в строке, набранного последним, осуществляется нажатием клавиши ЗБ.

2. **Стирание строки** может осуществляться с помощью клавиши ВК или команды DELETE. Нажатием клавиши ВК после набора соответствующего номера строки осуществляется стирание строки и ее номера, например 100 ВК. С помощью команды DELETE может осуществляться стирание строки, ряда строк, следующих друг за другом, и всей программы. Для стирания одной строки необходимо в команде указать номер строки. Например, команда DELETE 110 обеспечит стирание строки с номером 110; команда DELETE 10, 120 обеспечит стирание строк с 10-й по 120-ю; команда DELETE 1, 8191 обеспечит стирание всей программы.

3. **Получение текста** отредактированной программы выполняется с помощью команды LIST ВК. После вывода программы на экран выдается сообщение READY.

Например: команда LIST 65 выполняет вывод строки с номером 65; команда LIST 25, 120 выполняет вывод строк с 25-й по 120-ю включительно.

4. **Запуск программы** в косвенном режиме осуществляется командой RUN, начиная со строки с наименьшим номером. При использовании команды RUN для повторного прохождения программы все значения переменных стираются в памяти ЭВМ. Если в программу включена функция RND, то ее начальное значение восстанавливается.

5. **Останов выполнения программы** производится командой CY/P. При этом на экран выводится P, а затем READY. Команда не изменяет значение переменных.

6. **Изменение или исправление строки** может выполняться с помощью оператора перехода GOTO. Для этого необходимо ввести номер заменяемого оператора, а затем после GOTO номер нового оператора. Например, для замены оператора 20 READ A, B на оператор 45 READ C, D необходимо ввести оператор 20 GOTO 45.

Отладку программы удобно осуществлять при работе в непосредственном режиме. Для останова вычислений в любом месте программы может использоваться оператор STOP. После останова можно внести изменения. Для продолжения выполнения программы используется оператор GOTO *n* (*n* — номер строки оператора, стоящего за оператором STOP).

Не разрешается вносить изменения в цикл, образованный операторами FOR – NEXT, если произошел останов по оператору STOP, расположенному внутри цикла.

Сообщения пользователю. БЕЙСИК производит проверку операторов программы и вводимых данных и при обнаружении ошибок выдает сообщение об ошибке в виде строки ОШИБКА XXX СТРОКЕ YYY, где XXX – код ошибки, YYY – номер строки, в которой обнаружена ошибка. Если YYY = 0, то это означает, что ошибка произошла в режиме команд, т. е. в последней вводимой строке.

Ошибки подразделяются на неустранимые (коды 0–64) и устранимые коды (65–127). Появление неустранимой ошибки вызывает сообщение о ней и прекращение выполнения программы. Устранимая ошибка не вызывает прекращение выполнения программы.

Коды наиболее часто встречающихся ошибок приведены в табл. 2.

Таблица 2

Код ошибки	Причина ошибки
	Неустранимые ошибки
0	Переполнение памяти, отведенной пользователю
1	Нераспознаваемый оператор
2	Недопустимый оператор GOTO и GOSUB
3	Недопустимый знак, ограничивающий оператор и вызывающий преждевременное окончание его действия
4	Оператор RETURN без соответствующего оператора GOSUB
5	Неправильно сформирован индекс
6	Индекс не попадает в интервал от 0 до 255 или превышает максимум, установленный программой
7	Несоответствие скобок в операторе
8	Недопустимый оператор LET
9	Недопустимый знак отношения в операторе IF
10	Недопустимый оператор IF
11	Недопустимый оператор PRINT
12	Слишком длинная вводимая строка (80 знаков)
13	Неправильная размерность в операторе DIM
14	В памяти недостаточно места для массива
15	Неправильно сформирован оператор DEF
16	Недопустимый номер строки или значение размерности
17	Оператор DIM для ранее описанного или использованного элемента
18	Неправильная переменная в списке оператора INPUT
19	Неправильная переменная в списке оператора READ
20	Данные в списке оператора READ исчерпаны
21	Неправильный формат оператор DATA
22	Недопустимый оператор FOR
23	FOR не заканчивается соответствующим оператором NEXT
24	Оператор NEXT без оператора FOR
25	Несоответствие кавычек в операторе

Продолжение табл. 2

Код ошибки	Причина ошибки
26	Неправильно установлена функция
27	Неправильно сформировано выражение (пропущен порядок числа в формате *E*)
	Устранимые ошибки
120	Недопустимые знаки при вводе
121	Введено недостаточно данных по оператору INPUT
122	Введено слишком много данных по оператору INPUT
123	Несуществующая переменная
124	Число слишком велико для фиксации (вероятно комбинация индексов превышает диапазон)
125	Переполнение или заем при делении (умножении)
126	Квадратный корень из отрицательного числа
127	Логарифмы отрицательного числа или нуля; переполнение при вычислении EXP

ЭВМ "Искра-226"

Клавиатура ЭВМ. С помощью клавиатуры осуществляется ввод в память ЭВМ программ и данных, управление подготовкой и выполнением программ. Клавиатура разбита на восемь зон.

В первой зоне расположены клавиши с буквами русского и латинского алфавитов. Переход с русского на латинский регистр осуществляется с помощью переключателя, расположенного в левой части зоны. Клавиши первой зоны используются при вводе символов и ключевых слов. Переход на регистр ключевых слов осуществляется клавишами SHIFT и SHIFT LOCK. Для записи ключевого слова требуется нажатие на клавишу SHIFT, а затем на клавишу с нужным ключевым словом. Переход на регистр ключевых слов выполняется нажатием на клавишу SHIFT LOCK, снятие фиксации осуществляется нажатием на клавишу SHIFT. Клавиша CR/LF (возврат каретки/перевод строки) используется при завершении набора оператора. При нажатии на клавишу происходит передача управления системе программирования БЕЙСИК.

Во второй зоне размещены четыре клавиши: две для редактирования программы и две для управления счетом.

Клавиша LINE ERASE служит для стирания строки программы, а клавиша BACK SPACE – для стирания части строки путем многократного нажатия.

Для перехода на пошаговый режим выполнения программы необходимо нажать клавишу HALT/STEP и с помощью оператора GOTO 10 перейти в начало программы. Переход в автоматический режим выполняется нажатием на клавишу CONTINUE.

В третьей зоне расположены клавиши с цифрами. Они дублируют клавиши верхнего ряда первой зоны и не требуют перехода с регистра на регистр.

В четвертой зоне расположены шесть клавиш с наиболее часто используемыми ключевыми словами: RUN, CLEAR, LIST, PRINT, LOAD, SAVE.

В пятой зоне расположены клавиши со знаками арифметических операций и стандартных функций. Для перехода на стандартные функции требуется нажать клавишу SHIFT.

В шестой зоне расположены клавиши для редактирования программ: EDIT, RECALL, INSERT, DELETE, ERASE.

В седьмой зоне расположены клавиши для перемещения курсора. Они используются только в режиме редактирования, после нажатия клавиши EDIT. В правом верхнем углу расположена клавиша RESET, использующаяся для аварийного прекращения выполнения программы. После нажатия на клавишу управления передается пользователю.

Средства редактирования программ

Для нанесения исправлений в программы используются клавиши клавиатуры ЭВМ.

1. **Стирание символа** в набранной строке осуществляется нажатием клавиши BACK SPACE. Каждое нажатие клавиши стирает один символ.

2. **Исправления в программе в режиме редактирования.** Перевод ЭВМ в режим редактирования выполняется нажатием клавиши EDIT. В этом режиме разрешается перемещать курсор с помощью клавиш (↑, ← – –, ←, →, – – →, ↓), удалять символы внутри строки, стирать часть строки или всю строку, раздвигать строку для вставки символов.

Удаление символа после подвода курсора к нему выполняется нажатием клавиши DELETE. При этом происходит стирание отмеченного курсором символа, а текст строки, расположенный справа от курсора, сдвигается на одну позицию влево.

Внесение символа после подвода курсора к месту вставки осуществляется нажатием клавиши INSERT. Отмеченная часть строки сдвигается на одну позицию вправо, при этом освобождается место для вставки символа. При многократном нажатии на клавишу INSERT строка раздвигается на соответствующее число позиций.

Стирание части строки после подвода курсора к первому стираемому символу выполняется нажатием клавиши ERASE. При этом стирается часть строки от курсора и до ее конца.

Стирание всей строки осуществляется нажатием клавиши LINE ERASE.

3. **Исправление программы, хранящейся в оперативной памяти.** Вызов строки из оперативной памяти производится нажатием клавиш EDIT и RECALL. После этого на экране появляется текст нужной строки и далее выполняется редактирование.

Например, для вызова оператора с номером строки 40 следует выполнить процедуру

```
: 40
EDIT
* 40
```

RECALL
* 40 A = B + C

После редактирования нажимается клавиша CR/LF (возврат каретки—перевод строки) и строка записывается на старое место.

Добавление строки в программу выполняется путем набора номера строки, ее текста, а затем нажимается клавиша CR/LF.

Удаление строки из программы осуществляется после набора ее номера нажатием клавиши CR/LF

Отладка программ

После занесения набранной строки в память ЭВМ система программирования БЕЙСИК выполняет синтаксический контроль. При обнаружении ошибки на экран выводится сообщение с кодом ошибки.

Например, 20 AT =C/2
^ ERR 06

Стрелка в сообщении указывает вероятное место ошибки. Оператор следует исправить. Для визуального просмотра программы используется команда LISTS. По этой команде на экран дисплея выводится первая страница набранной программы (23) строки. Нажатие клавиши CR/LF вызывает вывод следующей страницы и так далее до конца программы. Обнаруженные ошибки исправляются в режиме редактирования.

Для отладки программ используется ряд операторов.

Оператор STOP имеет вид STOP < СИМВОЛЬНАЯ КОНСТАНТА >. Оператор вызывает останов выполнения программы и вывод на экран дисплея указанной в нем символьной константы. Пуск остановленной программы производится нажатием клавиши CONTINUE. После останова машины можно проверить значения интересующих переменных, а затем продолжить выполнение. В отлаженной программе включенные операторы STOP следует удалить.

Операторы TRACE и TRACE OFF позволяют проследить ход выполнения программы путем ее трассировки. Оператор TRACE начинает трассировку, а TRACE OFF заканчивает ее. Трассировка заключается в выводе на экран дисплея результатов выполнения операторов. Для удобства наблюдения вывод информации необходимо замедлить командой SELECK PK, где K = 0, 1, 2, ..., 9. Обычно достаточно взять K, равное 4–5. После выполнения оператора TRACE на экран выводится информация об изменении значений переменных, о передаче управления, о результатах вычислений. При наличии печатающего устройства результаты трассировки можно выдавать на печать.

Для этого необходимо переключить вывод информации с экрана дисплея на печать командой SELECT LISTOC в начале программы, а в конце выполнить обратный переход командой SELECT LIST05.

Помимо трассировки программы ход ее выполнения можно проследить в пошаговом режиме. Для перехода на этот режим после загрузки программы необходимо нажать клавишу HALT/STEP. Затем необходимо в режиме непосредственного счета обратиться в начало программы опе-

ратором GOTO 10. Пошаговое выполнение программы требует повторного нажатия на клавишу HALT/STEP. Для перехода в автоматический режим необходимо нажать клавишу GONTINUE.

Коды наиболее часто встречающихся ошибок приведены в табл. 3.

Таблица 3

Код ошибки	Причина ошибки
ERR01	Переполнение доступной пользователю оперативной памяти
ERR03	Ошибка при выполнении математических операций
ERR06	Синтаксическая ошибка в введенной с клавиатуры строке
ERR08	Не определена функция
ERR11	Неправильный номер строки
ERR12	Неправильный оператор
ERR15	Неправильный номер строки или диапазон строк
ERR18	Недопустимая величина (числа, размерности, массива, индекса)
ERR22	Неопределенный массив или переменная, встретившаяся в программе
ERR25	Недопустимая пара операторов FOR–NEXT или GOSUB–RETURN
ERR27	При выполнении оператора READ исчерпаны данные, определенные в операторе DATA
ERR29	Недопустимый формат данных, вводимых оператором
ERR43	Недопустимое присваивание
ERR45	Длина оператора больше 253 символов

Приложение 2

Вывод результатов в виде графиков, таблиц, гистограмм

Вывод результатов в виде графиков

Результаты, полученные на ЭВМ, целесообразно представлять в виде графиков, таблиц, гистограмм.

Для построения графика функции $y = f(x)$ при изменении аргумента x от x_0 до x_m с шагом Δx необходимо в каждой строке выводить некоторый символ, например "*", в позиции, соответствующей значению функции. Для определения позиции в строке, в которой должна быть помещена точка функции, используется функция TAB. При использовании этой функции в операторе

PRINT TAB (Y); "*"

символ "*" будет размещен в позиции, номер которой определяется значением переменной Y. При этом переменная Y округляется до ближайшего целого, не превосходящего значения Y. Если необходимо округлить Y до ближайшего целого, то следует увеличить его на 0,5, т.е. Y + 0,5. Перед печатью следующей точки функции осуществляется переход на следующую строку.

Таким образом, координаты точки функции определяются дискретными значениями с минимальным шагом, равным по горизонтали одной позиции в строке, а по вертикали – интервалу между строками. Поле экрана,

в котором может быть размещен график, не должно превосходить по вертикали 21 строку, а по горизонтали – 72 позиции. Поэтому, чтобы график был наглядным, необходимо выбирать масштабы как по вертикали, так и по горизонтали.

Поскольку при каждом выполнении оператора PRINT осуществляется переход на следующую строку, т.е. номер строки изменяется с постоянным шагом, равным, например 1, то удобно по вертикали задавать изменение аргумента, тогда по горизонтали будут откладываться значения функции. График функций будет повернут на 90° по часовой стрелке относительно привычного в математике положения.

Для получения на экране графика целиком необходимо, чтобы аргумент имел не более 20 значений (по количеству строк). Шаг изменения аргумента определяется отношением

$$\Delta x \geqslant] \frac{x_m - x_0}{20} + 1 [.$$

Для выбора масштаба по горизонтали (по переменной Y) необходимо знать максимальное y_{max} и минимальное y_{min} значения функции.

Если функция имеет только положительные значения, то должно удовлетворяться условие $My_{max} \leqslant N$, где M– масштабный множитель, N – количество позиций в строке, отведенных для построения графика от оси x до y_{max}. Тогда положение точки функции определяется значением My_i.

Если функция имеет только отрицательные значения, то должно удовлетворяться условие $M | y_{min} | \leqslant N$, где N – количество позиций в строке, отведенных для построения графика от оси x до y_{min}. Поскольку ось x в этом случае расположится справа, а номер позиции отрицательным быть не может, то необходимо ввести смещение S, равное N.

Тогда положение точки функции будет определяться значением

$$My_i + N.$$

Если функция имеет положительное и отрицательное значения, то определение масштаба осуществляется из выражения

$$M | y_{min} | + My_{max} \leqslant N.$$

Если ось x расположена в центре графика, то смещение ее относительно левого края поля равно $N/2$. Тогда положение точки функции на строке определяется выражением

$$My_i + N/2.$$

Положение этой точки на строке будет определяться функцией TAB (M * Y).

Если переменная Y имеет отрицательные и положительные значения, то необходимо задавать смещение S для графика, так как функция TAB может иметь только положительные значения в интервале от 0 до 71. В этом случае максимальное (A_{max}) и минимальное (A_{min}) значения функции TAB определяются из выражений $A_{max} = My_{max}$ и $A_{min} = M | y_{min} | + S$.

П. 33. Построить график функции $y = \cos x$ при изменении аргумента x

от 0 до 2 π с шагом $\pi/8$. Для построения этого графика потребуется n строк, которое определяется по формуле

$$n =] \frac{x_{max} - x_{min}}{\Delta x} + 1 [.$$

В рассматриваемом примере $n = 17$.

Функция y является симметричной относительно оси x и имеет значения $y_{max} = 1$ и $y_{min} = -1$. Выбрав для построения графика 60 позиций, т.е. $A_{max} = 60, A_{min} = 0$, получим $S = 30$, а $M = \frac{A_{max} - 30}{y_{max}} = 30$. Используя в цикле оператор

PRINT TAB (Y * 30 + 30); "*"

можно вывести на экран точки графика этой функции.

Для большей наглядности графика целесообразно выводить его вместе с осями координат.

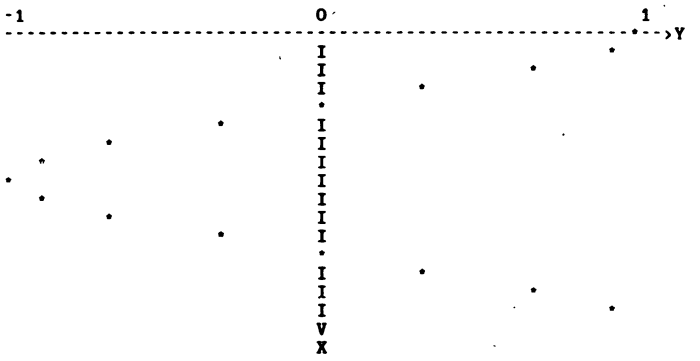
Функция $y = \cos(x)$ не вычисляется для отрицательных значений аргумента x, поэтому в первой строке следует разместить ось y, например, используя в цикле оператор

PRINT "–";

Символ ";" в конце оператора блокирует переход на новую строку при каждом выполнении оператора. В этой же строке в 60-й позиции должно быть помещено значение функции y при $x = 0$, т.е. символ "*". Ось абсцисс (ось x) должна располагаться в 30-й позиции и обозначается с помощью символа "I". При выводе на экран символа "I", обозначающего ось x, и символа "*", обозначающего точку функции, необходимо учитывать следующее. Если значение функции y в данной точке положительно, то сначала в 30-й позиции необходимо расположить символ "I", а затем в позиции с номером $M \cdot y + 30$ – символ *. Для отрицательных значений функции y, наоборот, сначала символ "*", а затем символ "I".

Учитывая сказанное выше, а также вводя обозначение осей x и y и стрелок, получим программу и результаты ее выполнения, представленные в следующем виде:

```
05 OPEN "LP:" FOR OUTPUT AS FILE #1
10 PRINT #1,"-1";TAB(30);"0";TAB(60);"1"
20 FOR I=1 TO 59\PRINT #1,"-";\NEXT I
30 PRINT #1,"*";\PRINT #1,"-->Y"
40 H=PI/8
50 FOR X=H TO 2*PI STEP H
60 Y=30*COS(X)
70 IF ABS(Y)<1.00000E-04 THEN 130
80 IF Y>0 THEN 110
90 PRINT #1,TAB(Y+30);"*";TAB(30);"I"
100 GO TO 140
110 PRINT #1,TAB(30);"I";TAB(Y+30);"*"
120 GO TO 140
130 PRINT #1,TAB(30);"*"
140 NEXT X
150 PRINT #1,TAB(30);"V"
160 PRINT #1,TAB(30);"X"
170 CLOSE #1
180 END
```



Оператор строки 10 выводит значения ординат – 1, 0, + 1.

Операторы строк 20 и 30 выводят 59 символов "–" (ось y), в 60-й позиции – символ "*" (значение первой точки графика), а далее выводят обозначение стрелки и символ „Y".

Оператор строки 70 проверяет не лежит ли точка на оси x. Если точка находится на оси x, то выполняется оператор строки 130, выводящий символ "*". Оператор строки 90 выводит для отрицательных значений переменной Y сначала символ "*", затем символ "I", обозначающий ось x.

Оператор строки 110 для положительных значений переменной Y выводит сначала символ "I", а затем символ "*". Операторы строк 150 и 160 выводят обозначение стрелки и символ "X".

В приведенной и последующих программах включены команды OPEN и CLOSE, а также номера каналов (# 1) в операторах PRINT. Назначение команд и номеров каналов будет рассмотрено ниже.

Вывод результатов в виде таблиц

При выводе результатов в виде таблиц целесообразно выводить заголовок и шапку таблицы, в которой указываются имена переменных, подлежащих выводу. Удобно разграничивать таблицы горизонтальными и вертикальными линиями. Для этих целей используются оператор PRINT и функция TAB.

П.34. Вывести на печать значения аргумента x и функций $y_1 = \dfrac{\sin x}{x}$ и $y_2 = \dfrac{\cos x}{x}$, если значения x изменяются от 0 до π с шагом $\pi/10$.

Таблицу оформим следующим образом. В первой строке – заголовок таблицы "Таблица значений функции y_1 и y_2". Во второй и четвертой строках – горизонтальные линии. В третьей строке в позиции *1* – вертикальная линия, в позиции *3* – символ "X", в позиции *8* – вертикальная линия, в позициях *11* и *12* – символы "Y1", в позиции *18* – вертикальная линия, в позициях *21* и *22* – символы "Y2", в позиции *28* – вертикальная линия. В строках с пятой по пятнадцатую следует расположить значения аргумента и функций, а в шестнадцатой – горизонтальную линию.

Программа и результаты в виде таблицы имеют вид:

```
05 OPEN "LP:" FOR OUTPUT AS FILE #1
10 PRINT #1,TAB(5);"ТАБЛИЦА ЗНАЧЕНИЙ ФУНКЦИЙ Y1 И Y2"
20 FOR I=1 TO 28\PRINT#1,"-";\NEXT I\PRINT#1
30 PRINT #1,TAB(3);"X";TAB(8);"I";TAB(11);"Y1";TAB(18);"I";
40 PRINT #1,TAB(21);"Y2";TAB(28);"I"
50 FOR I=1 TO 28\PRINT #1,"-";\NEXT I\PRINT #1
60 FOR X=.1 TO 1 STEP.1
70 Y1=SIN(X)/X \ Y2=COS(X)/X
80 PRINT #1,"I";TAB(3);X;TAB(8);"I";TAB(10);Y1;TAB(18);"I";
90 PRINT #1,TAB(20);Y2;TAB(28);"I"
100 NEXT X
110 FOR K=1 TO 28 \ PRINT #1,"-"; \ NEXT K
115 PRINT #1," "
120 CLOSE #1
130 END
```

ТАБЛИЦА ЗНАЧЕНИЙ ФУНКЦИЙ Y1 И Y2

X	I	Y1	I	Y2	I
.1	I	.998334	I	9.95004	I
.2	I	.993347	I	4.90033	I
.3	I	.985067	I	3.18446	I
.4	I	.973546	I	2.30265	I
.5	I	.958851	I	1.75517	I
.6	I	.941071	I	1.37556	I
.7	I	.920311	I	1.09263	I
.8	I	.896695	I	.870883	I
.9	I	.870363	I	.690678	I
1	I	.841471	I	.540302	I

Построение гистограмм

Гистограмма представляет собой изображение отрезков, длины которых пропорциональны соответствующему значению функции. Отрезки можно изображать в виде последовательности символов, например, "*", в количестве, пропорциональном значению функции. Поскольку поле экрана имеет ограниченное число позиций, необходимо прибегать к изменению масштаба, который определяется по формуле

$$M = \left] \frac{N}{y_{\max}} + 0{,}5 \right[,$$

где N – количество позиций в строке; $y_{\max}$ – максимальное значение функции.

П.35. Значения функции заданы в массиве Y, состоящим из 10 элементов. Вывести на печать значения массива в виде гистограммы. Ширина поля, отведенного для построения гистограммы, равна 50 позициям.

Количество строк в гистограмме определяется количеством элементов массива Y.

Для выбора масштаба необходимо найти наибольший элемент массива Y.

Гистограмму удобно оформлять следующим образом.

В позиции *1* выведем обозначения элементов массива Y; в позиции *3* – символ "I" для обозначения вертикальной линии, в позициях *4–54* расположим последовательность символов "*" в количестве, пропорциональном

величине y_i; начиная с позиции *56*, выведем численное значение y_i (элемента массива). Вывод линий гистограммы оформим в виде подпрограммы.

Программа и результат ее выполнения имеют вид:

```
10 DIMY(10)
15 OPEN "LP:" FOR OUTPUT AS FILE #1
30 FOR I=1 TO 10
40 INPUT Y(I)
50 IF Y(I)>Y1 THEN Y1=Y(I)
60 NEXT I
70 M=INT(50/Y1+.5)
80 FOR I=1 TO 10
90 PRINT #1,"Y";I;TAB(3);"I";
100 Y2=Y(I)+M
110 GOSUB 140
120 NEXT I
125 CLOSE #1
130 STOP
140 REM ПОДПРОГРАММА
150 FOR K=1 TO Y2
160 PRINT #1,TAB(6);"·";
170 NEXT K
180 PRINT #1,TAB(56);Y(I)
190 RETURN
200 END
```

```
Y 1 I ··········                                  9
Y 2 I ·******                                     6
Y 3 I ···········                                 11
Y 4 I ····················                        20
Y 5 I ···············                             15
Y 6 I ·····························               30
Y 7 I ··························                  28
Y 8 I ·····················                       24
Y 9 I ···········                                 12
Y 10 I·********                                   7
```

Обработка символьных переменных

Символьной переменной является переменная, значением которой является последовательность символов (символьная строка). Символьные переменные используются при решении задач невычислительного характера.

Символьная переменная обозначается именем (как и обычные переменные) с добавлением в конце имени символа "¤" (знак денежной единицы). Например, X¤, Y1¤. Количество символов в символьной строке не должно превосходить 255. Для записи символьных строк могут использоваться любые символы языка БЕЙСИК. При записи в программе символьная строка заключается в кавычки.

Допускается использование символьных массивов, которые описываются так же, как обычные массивы. Например, оператор DIMX¤(5,20), S(6) описывает символьную матрицу X размером 6×21 и числовой массив *S*, состоящий из семи элементов. Элементы символьных массивов обозначаются в виде переменных с индексами. Например, X¤(2,5) – элемент 3-й строки, 6-го столбца символьной матрицы X.

Значения символьным переменным задают двумя способами. Первый способ – с помощью операторов ввода INPUR или READ. В операторе DATA строка символов заключается в кавычки, например,

```
10 READ X¤,S
20 DATA; "РАСХОД", 0
```

Операторы осуществляют ввод символьной переменной X¤. которая получает значение РАСХОД, и числовой переменной S, которая получит значение 0.

Второй способ – с помощью оператора присваивания, например, X¤ = = "РАСХОД".

Над символьными переменными допускается только одна операция – операция сочленения, которую обозначают символом & (амперсанд), например, A¤ & B¤.

При выполнении этой операции значения двух символьных переменных объединяются в одно. Так, если A¤ = "РАС", а B¤ = "ХОД", то после выполнения операции выражение A¤&B¤ будет иметь значение РАСХОД.

Для символьных переменных используются следующие стандартные функции.

ASC (X¤) – предназначена для перевода двоичного кода аргумента в десятичное число. Аргумент может быть представлен только одним символом. Например, если символьная переменная X¤ имеет значение "А", то после выполнения функции ASC (X¤) ее значение будет равно 65.

LEN(X¤) – предназначена для нахождения длины символьной переменной, т.е. количества входящих в нее символов, включая пробелы. Например, если символьная переменная X¤ имеет значение "РАСХОД ПАРА", то после выполнения функции LEN(X¤) ее значение будет равно 11.

POS (X¤, Y¤ Z – предназначена для определения позиции первого появления последовательности символов Y¤ в символьной переменной X¤, начиная с позиции Z. Например, если переменная X¤ имеет значение РАСХОД ПАРА, то после выполнения функции POS (X¤, "А",5) ее значение будет равно 9.

Если последовательность символов не найдена или значение переменной X¤ является пустой строкой, то выдается значение 0, а если последовательность символов Y¤ является пустой строкой, то выдается номер позиции, соответствующий Z.

SEG¤ (X¤, Y, Z) – предназначена для выделения из значения переменной X¤ последовательности символов, начиная с позиции Y, кончая позицией Z.

Например, если переменная X¤ имеет значение РАСХОД ПАРА, то функция SEG¤ (X¤, 1,6) имеет значение РАСХОД. Если номер позиции задач Y ⩽ 0, то просмотр начинается с первой позиции. Если задано Z ⩽ 0 или Y > LEN (X¤), то выдается пустая строка. Если задано Z > LEN(X¤), то выдается длина строки.

STR¤ (X) – предназначена для преобразования значения аргумента из числовой формы в символьную. Например, если X = 10, то результатом выполнения функции STR¤ является символьная строка "10".

TRM¤ (X¤) – предназначена для удаления конечных пробелов.

VAL (X¤) – предназначена для получения числа, представленного знаковой строкой. Если строка не является числом, то выдается сообщение

об ошибке. Например, если X$ = "10", то значение функции Y = 1 + + VAL (X$) будет равно 11.

П.36. Составить программу для определения среднего балла группы, среднего балла каждого учащегося и среднего балла каждой из пяти дисциплин по результатам сессии. Программа и результат ее выполнения имеет вид:

```
1 OPEN "LP:" FOR OUTPUT AS FILE #1
10 DIM A$(10,6)
20 S0=0
30 FOR I=1 TO 9
40 PRINT "ВВЕДИТЕ ФАМИЛИЮ УЧАЩЕГОСЯ С НОМЕРОМ I=";I
50 INPUT A$(I,0)
60 S=0
70 FOR J=1 TO 5
80 PRINT "ВВЕДИТЕ ОЦЕНКУ УЧАЩЕГОСЯ С НОМЕРОМ I=";I
90 INPUT A$(I,J)
100 S=S+VAL(A$(I,J))
110 NEXT J
120 S0=S0+S
130 A$(I,6)=STR$(S/5)
140 NEXT I
150 A$(10,6)=STR$(S0/45)
160 FOR J=1 TO 5
170 S1=0
180 FOR I=1 TO 9
190 S1=S1+VAL(A$(I,J))
200 NEXT I
210 A$(10,J)=STR$(S1/9)
220 NEXT J
230 PRINT #1,TAB(25);"РЕЗУЛЬТАТЫ ЭКЗАМЕНОВ"
240 PRINT #1,TAB(15);"МАТ. ФИЗ. ХИМ. ИСТ. ТМЕХ. СР.БАЛЛ"
250 FOR I=1 TO 9
260 PRINT #1,A$(I,0);
270 FOR J=1 TO 6
280 PRINT #1, TAB(9+6*J);A$(I,J);"  ";
290 NEXT J
300 PRINT #1," "
310 NEXT I
320 PRINT #1,"СР.БАЛЛ";TAB(14);" ";
330 FOR J=1 TO 6
340 PRINT #1,A$(10,J);" ";
350 NEXT J
355 PRINT #1," "
360 CLOSE #1
370 END
```

```
          РЕЗУЛЬТАТЫ ЭКЗАМЕНОВ
         МАТ.      ФИЗ.      ХИМ.      ИСТ.      ТМЕХ.      СР.БАЛЛ
АНДРЕЕВ  4         5         5         4         5          4.6
БОРИСОВ  4         4         5         3         3          3.8
ВЛАСОВ   5         5         5         5         5          5
ГРОМОВ   4         4         4         5         4          4.2
ДУБОВ    3         3         3         3         3          3
ЗОРИН    4         5         3         4         3          3.8
ИВАНОВ   4         4         4         4         5          4.2
ПЕТРОВ   5         5         5         5         5          5
СИДОРОВ  4         4         4         4         4          4
СР.БАЛЛ  4.11111 4.33333 4.22222 4.11111 4.11111 4.17778
```

Результаты сдачи экзаменов заносят в символьную матрицу A$. В нулевом столбце матрицы записывают фамилию учащегося, в столбцы с 1-го по 5-ый заносят оценки, шестой столбец резервируют для среднего балла учащегося. Ввод исходной информации осуществляется оператором INPUT. Нулевая строка матрицы не используется. Оператор в строке 100 накапливает сумму баллов учащегося. Так как элемент массива A$ явля-

ется строковой величиной, то для преобразования ее в арифметическую используется функция VAL. Средний балл учащегося записывают в шестой столбец матрицы A$ с помощью оператора в строке 130. Так как средний балл является арифметической величиной, то перед записью ее в массив A$ она преобразуется в строковую величину с помощью функции STR$. Средний балл группы записывают в шестой столбец 26-й строки с помощью оператора в строке 150. В следующем двойном цикле вычисляются средние баллы по каждой дисциплине, которые записываются в 26-й строке в столбцах с 1-го по 5-й.

Операторы строк с 230 по 350 осуществляют печать результатов в виде таблицы.

Запись программы на внешнее запоминающее устройство

В случае, если необходимо сохранить программу для следующего сеанса работы с ЭВМ, ее можно записать на внешнее запоминающее устройство – накопитель на гибких магнитных дисках (ГМД).

Для этой цели используется команда

SAVE <ИМЯ>,

где <ИМЯ> программы состоит из набора от одного до шести символов, начинающегося с буквы.

Если программе не было присвоено имя, она получает стандартное имя NONAME. Если необходимо присвоить новое имя программе, находящейся в оперативной памяти, то это осуществляется с помощью команды

RENAME <ИМЯ>

Загрузка программы, хранящейся на ГМД, в оперативную память осуществляется командой

OLD <ИМЯ>

а стирание с диска производят командой

UNSAVE <ИМЯ>

Если необходимо старую версию программы заменить на новую, то используют команду

REPLACE <ИМЯ>

При этом старая версия программы стирается.

Для объединения нескольких программ в одну используется команда APPEND <ИМЯ>. При выполнении этой команды программа с именем, указанным в команде, объединяется с программой, находящейся в оперативной памяти машины. Для объединения необходимо, чтобы в обоих программах были строки с разными номерами, так как в противном случае строки программы с именем, указанном в команде APPEND, будут заменены строками с теми же номерами из оперативной памяти. Объединенная программа упорядочивается в соответствии с номерами строк операторов.

Команда SCR используется для очистки оперативной памяти от имеющейся в ней программы, удаляет ее имя и устанавливает стандартное имя NONAME. Команда LENGTH определяет длину введенной программы в словах.

Вывод данных на печатающее устройство

Перед выводом результатов решений на печать они должны располагаться в отведенном для них месте (файле). Для объявления (открытия) файла используется команда OPEN.

Эта команда должна предшествовать любым действиям над файлами. В команде указывается имя файла, номер канала, дано сообщение о том, какие действия будут выполняться (ввод данных из файла в оперативную память или их вывод из оперативной памяти в файл). Вводу соответствуют ключевые слова FOR INPUT, а выводу – FOR OUTPUT. Общий вид команды:

OPEN < имя файла > $\left\{\begin{matrix}\text{FOR INPUT}\\ \text{FOR OUTRUT}\end{matrix}\right\}$ AS FILE # < номер канала >

Номер канала – целое число от 1 до 12.

При открытии файла для вывода данных из оперативной памяти на печать имя файла должно быть стандартным LP:

Например, команда

```
5 OPEN LP : AS FILE # 1
```

связывает канал 1 с печатающим устройством.

Для вывода результатов на печатающее устройство необходимо использовать оператор 20 PRINT # 1A; B, C.

Команда CLOSE закрывает файл и выделяет на диске место для его хранения. В противном случае файл не сохраняется после окончания работы с машиной.

Например, команда

```
70 CLOSE ␣ # 1
```

закрывает файл, открытый командой OPEN.

В программах, представленных в приложении *2*, результаты выводятся на печатающее устройство. Если результаты необходимо вывести на экран дисплея, то из программ следует удалить команды OPEN и CLOSE, а из операторов PRINT – номера каналов (#1).

Приложение 3

В современных персональных ЭВМ широко используется язык БЕЙСИК-А, предоставляющий пользователю гораздо большие возможности по сравнению с рассмотренным в книге языком.

Константы. Допускается использование констант следующих типов:

целые, действительные, действительные удвоенной точности, символьные, шестнадцатеричные и восьмеричные.

Целые константы представляют собой последовательность цифр со знаком. Знак плюс можно опускать, например 17, −125, +49.

Действительные константы могут записываться в форме с десятичной точкой и в форме с порядком. Для обозначения основания десятичной системы счисления используется буква E. Например, 0,525, −17.4, 1.5E −12, .3E8, 1.

Незначащие нули можно опускать. При записи действительных констант можно использовать символ "!". Например, −1! означает действительную константу, равную минус единице.

Константы удвоенной точности записываются в форме с порядком. В этом случае основание системы счисления обозначается буквой D, например −2.5D17.

При записи констант удвоенной точности можно использовать символ "#", например −1 # означает константу удвоенной точности, равную минус единице.

Константы действительные и удвоенной точности представляются в машине приближенно. При этом константы удвоенной точности представляются более точно за счет большего числа разрядов, отведенных для их хранения.

Символьные константы представляют собой последовательность любых символов языка, заключенную в кавычки. Например, "NOMER", "ЗНАЧЕНИЕ ФУНКЦИИ".

Восьмеричные и шестнадцатеричные константы представляют собой число в восьмеричной и шестнадцатеричной системах счисления. При записи этих констант перед ними записываются символы &O – для восьмеричной и &H – для шестнадцатеричной константы.

Переменные. Для обозначения переменных может использоваться имя, состоящее из буквенных или комбинации буквенных и цифровых символов. Имя начинается обязательно с буквы и содержит до 40 символов. Переменные могут принимать значения констант любого типа. Для обозначения типа переменных используются следующие символы: "%" – для переменной целого типа; "!" – для переменной действительного типа; "#" – для переменной удвоенной точности; "¤" – для символьной переменной. Указанные символы записываются после имени переменной, а при их отсутствии переменная считается действительной. Например, X%, A!, E#, FAM¤.

В языке допускается использование многомерных массивов. Количество измерений массивов не должно превосходить 255. Например, A(I, K, L) – элемент трехмерного массива действительного типа.

Тип массивов и их элементов указывается так же, что и для простых переменных, например B¤(5,4) – элемент матрицы B символьного типа.

Для описания типов переменных и массивов можно использовать специальный оператор DEF, в котором задается с помощью следующих символов: INT – целый; SGN – действительный; DBL – удвоенной точности; STR – символьный. Далее в операторе записываются буквы или последовательности букв, разделенные запятыми или тире, например оператор DEF

DBL A, H–K описывает тип переменных, имена которых начинаются с букв A, H, I, J, K как переменные удвоенной точности.

Функции. Кроме арифметических функций, которые уже были рассмотрены в книге, язык содержит ряд специальных функций, предназначенных для преобразования данных из одного типа в другой, функции для работы с символьными данными и др.

Некоторые из этих функций приведены при рассмотрении соответствующих разделов.

Операторы. Кроме операторов, рассмотренных выше, в языке используется ряд дополнительных операторов.

Оператор выбора. Структура операторов выбора следующая: ON <арифметическое выражение> GOTO <последовательность меток>. Например, ON K + 1 GOTO 20, 70, 90. Этот оператор в зависимости от значения выражения осуществляет переход к одной из указанных в нем меток. Так, если K + 1 = 2, то будет осуществлен переход к строке с номером 70, а если K + 1 = 3, то к строке с номером 90. Если значение выражения меньше 1 или больше 3, то выполняется оператор, следующий за оператором ON–GOTO.

Этот оператор удобно использовать для организации разветвлений в программе. При этом количество ветвей, равное количеству меток, указанных в операторе, может быть любым.

Условный оператор. Условный оператор может иметь структуру:

IF <условие> THEN<последовательность операторов 1>
ELSE <последовательность операторов 2>

Если условие имеет значение "истина", то выполняется<последовательность операторов 1>, в противном случае выполняется <последовательность операторов 2>. После выполнения операторов любой последовательности выполняется оператор, расположенный в следующей строке. Например, оператор

```
80 IF X > O THEN S1 = S1+X: K1=K1+1 ELSES2=S2+X:K2=K2+1
```

вычисляет значения переменных S1 и K1, если X>0, или S2 и K2 – в противном случае. Если последовательность операторов содержит несколько операторов, то они разделяются с помощью символа ":".

Операторы ввода–вывода. В операторе INPUT можно указывать символьную константу, заключенную в кавычки. При выполнении этого оператора выводится символьная константа, затем символ "?", и машина ожидает ввода значений переменных, указанных в списке ввода. Например, при выполнении оператора 10 INPUT "ВВЕДИТЕ N ="; N будет выведена следующая информация:

```
ВВЕДИТЕ N =

?
```

Для ввода значений символьных переменных можно использовать оператор LINE INPUT. При этом значения символьных переменных заключать в кавычки нет надобности. В отличие от оператора INPUT символьная константа может содержать символ "запятая".

Оператор PRINT USUNG, используется для вывода результатов в заданном формате. Структура оператора:

PTINT USING <список форматов>; <список вывода>,

где <список форматов> – последовательность специальных символов, заключенных в кавычки; <список вывода> – список переменных и элементов массивов, подлежащих вводу.

В качестве специальных символов используются следующие:

! – вывод только первого символа строки;
& – вывод всей строки;
\n␣\ – вывод n + 2 первых символов строки;
– вывод одной цифры;
. – вывод десятичной точки;
, – вывод запятой через каждые три цифры слева от десятичной точки;
+ – вывод числа со знаком /+ или –/;
– – вывод числа со знаком –;
* – вывод с заполнением пустых позиций символами "*";
^^^^ – вывод с выделением позиций для размещения символа E и двухразрядного порядка со знаком /для чисел в форме с порядком/.

Примеры использования форматов следующие:

Формат	Выводимое число	Результат
##	12	12
####.##	12	12.00
###.##	123	123.00
####,.	1234	1,234.
+ ###	–123	–123
##.##^^^^	123	1.23E+02

Оператор PRINT USING "+ ##.#"; A, B, C выводит значения переменных A, B, C вместе со знаками. Каждое значение имеет две цифры в целой части и одну цифру в дробной.

Для вывода данных можно использовать оператор WRITE, имеющий такую же структуру, что и оператор PRINT. Выводимые значения при использовании оператора WRITE отделяются друг от друга запятыми, а символьные данные заключаются в кавычки. Например, оператор WRITE A; T⌧, B выводит данные в следующем виде: 1.5E–30, "ИВАНОВ" ,–1.4.

Операторы PRINT, PRINT USING, WRITE выводят результаты на экран дисплея. Для вывода результатов на печатающее устройство следует использовать операторы LPRINT и LPRINT USING, которые имеют такую же структуру, что и операторы PRINT и PRINT USING.

Оператор цикла WHILE – WEND. Для организации цикла кроме оператора FOR–NEXT можно использовать оператор WHILE – WEND. Этот оператор используется для организации итерационного цикла. Структура

цикла при использовании для его организации оператора WHILE – WEND следующая:

<метка> WHILE <логическое выражение>
<операторы, входящие в цикл>
<метка> WEND

Например, оператор WHILE X < = 1 открывает цикл, в котором проверяется условие повторения цикла X < = 1. Закон изменения параметра цикла в этом операторе не указывается, поэтому перед циклом необходимо задать начальное значение параметра цикла с помощью оператора присваивания, а внутри цикла с помощью другого оператора присваивания изменять значение параметра цикла. Цикл повторяется до тех пор, пока выполняется условие X < = 1. Как только это условие не будет выполнено, то осуществляется переход к оператору, следующему за оператором WEND, обозначающим конец цикла.

Оператор SWAP X1, X2 позволяет поменять местами аргументы X1 и X2. Выполнение этого оператора равносильно выполнению трех операторов присваивания: A = X1 : X1 = X2 : X2 = A.

Работа с символьными данными. Над символьными данными допускается одна операция – сочленение (обозначаемая символом "+"). Например, если символьная переменная F$ имеет значение "ИВАНОВ", а переменная IO$ = "A.A", то в результате выполнения оператора присваивания FIO$ = F $ + IO$ символьная переменная FIO$ получит значение ИВАНОВ A.A.

Над символьными данными допускается операция отношения, например в условном операторе 40 IF A$ = "END" THEN 90 проверяют, является ли значение переменной A константой END. При сравнении символьных данных проверяется совпадение кодов ASCII символов, являющихся значением сравниваемых переменных или констант. Например, условие "PAB" > "PAA" всегда выполняется, так как код символа B больше кода символа A, а условие "PAB" > "PAC" никогда не выполняется по той же причине.

При работе с символьными данными используются специальные функции.

Длина символьных данных не должна превосходить 255 символов.

Функция LEN(X$) определяет длину символьной строки (количество символов в строке), являющейся значением аргумента X$. При определении длины строки учитываются все символы, в том числе и символы пробел. Например, оператор K = LEN ("ИВАНОВ A.A.") присвоит переменной K значение, равное 10.

Функция ASC(X$) определяет внутренний код ASCII символа, являющегося аргументом. Например, оператор K = ASC("IWANOW") присвоит переменной K значение, равное 73 – код символа I.

Функция INSTR(K, X$, Y$) определяет начальную позицию вхождения в строку, заданную аргументом X$, подстроки, заданной аргументом Y$. Необязательный параметр K – номер позиции в строке X$, с которой начинается поиск вхождения. Если K отсутствует, то считается равным 1. Например, оператор M = INSTR(X$, Y$), если X$ = "ABCABDE" и

Y$ = "AB", присвоит переменной M значение, равное 1, а оператор M = = INSTR(K, X$, Y$) при тех же значениях аргументов X$, Y$ и при K = 3 присвоит переменной M значение, равное 4.

Функция VAL(X$) определяет числовое значение аргумента, заданного в символьной форме. Например, оператор K = VAL(X$) при X$ = = "1988 ГОД" присвоит переменной K значение 1988.

Функция STR$(X) преобразует числовой аргумент в символьную форму. Например, оператор E$ = STR$(X), если X = 1988, переменной E$ присвоит значение символьной константы "1988".

Функция CHR$(K) преобразует аргумент, представленный в коде ASCII, в символ. Например, оператор C$ = CHR$(65) присвоит переменной C$ символ A.

Функция SPACE$(N) позволяет получить строку, состоящую из N пробелов, где N – целое число или переменная.

Функция STRING(N, M) позволяет получить N одинаковых символов, код ASCII которых равен M. Например, оператор PRINT SPACE$(10); STRING (20, 42) выведет на печать сначала 10 пробелов, а затем 20 символов "*", код которых равен 42.

Функция LEFT$(X$, N) выделяет из символьной строки X$ N левых символов, а функция RIGHT$(X$, N) – N правых символов. Например, оператор A$ = LEFT$(X$, 3) + "–" +RIGHT$(X$, 4) выделит из строки X$, имеющей значение "ГОД РОЖДЕНИЯ 1988", константу "ГОД", константу "1988" и объединит их, включив между ними символ "–", т.е. переменная A$ получит значение "ГОД–1988".

Функция MID$(X$,N, M) выделяет из строки X$ M последовательных символов, начиная с позиции N. Например, оператор A$ = = MID$(X$,4,4) при X$ = "ГОД 1988 МЕСЯЦ МАЙ" выделит и присвоит переменной A$ символы "1988".

Функция INKEY$ используется для ввода одного символа с клавиатуры дисплея. Например, оператор Q$ = INKEY$ присвоит переменной Q$ значение введенного с клавиатуры символа.

Функции MKI$, MKS$, MKD$ преобразуют аргумент, представленный в виде чисел (целого, действительного, действительного удвоенной точности) в символьную строку, состоящую из двух, четырех, восьми символов соответственно, а функции CVI, CVS, CVD – из символьной формы в числовую (соответственно целую, действительную или удвоенной точности).

Графические средства языка. *Оператор* SCREEN используется для установки режимов экрана. Аргумент, задающий режим работы экрана, может принимать следующие значения: 0 – текстовой режим, 1 – графический режим среднего разрешения (200 строк по 320 точек в строке), 2 – графический режим высокого разрешения (200 строк по 640 позиций в строке). Например, оператор SCREEN 2 устанавливает графический режим высокого разрешения.

Оператор CLS используется для очистки экрана.

Оператор LINE[X1, Y1)] – (X2, Y2)[,цвет] [B[F] [, маска]] используется для рисования линий и прямоугольников, где X1, Y1 и X2, Y2 – координаты точек; цвет –целое число, указывающее номер цвета; B – признак, используемый для вычерчивания прямоугольника (при этом X1, Y1 – ко-

ординаты верхней левой точки прямоугольника, а X2, Y2 – координаты нижней правой точки прямоугольника); F – признак, обеспечивающий заполнение прямоугольника цветом; маска – шестнадцатеричное число, используемое для рисования прерывистых линий. Например, оператор LINE (0,0)–(200, 100), 1,B нарисует на экране прямоугольник со сторонами 200 на 100 точек, а оператор LINE (0,0)–(200, 100) нарисует прямую линию, соединяющую точки с координатами (0,0) и (200, 100).

Аргументы, указанные в прямоугольных скобка, могут опускаться.

Оператор CIRCLE (X, Y) R [, <цвет> [<начало, конец> [<соотношение>]]] используется для изображения на экране окружностей и эллипсов, где X, Y – координаты центра, R – радиус, <начало, конец> – углы начала и конца рисования эллипса, <соотношение> – соотношение между осями эллипса. Например, оператор CIRCLE (200, 100) 50 нарисует на экране окружность радиуса 50, центр которой находится в точке с координатами (200, 100).

Оператор PSET(X, Y) [,<цвет>] используется для изображения точки в заданном месте экрана, где X, Y – координаты изображаемой точки, <цвет> – целое число, указывающее номер цвета. Например, оператор PSET(X*MX, SIN(X)*MY), размещенный в цикле, изменяющем параметр X, изобразит на экране синусоиду, где MX и MY – масштабы по осям X и Y, которые позволяют вывести график функции в поле выбранного размера.

Оператор LOCATE [Y] [, X] [, K] [,A1] [,A2] используется для перемещения курсора в заданное место экрана, где Y – номер строки, в которую следует переместить курсор (число от 0 до 25); X – номер позиции в строке, в которую следует переместить курсор; K – переменная, принимающая значение 0, если курсор не должен быть виден, или 1 – если курсор должен быть виден; A1 и A2 – соответственно начало и конец курсора, определяющие его размер (число от 0 до 31).

Работа с файлами данных. Результаты, получаемые при выполнении программы, выводятся на экран дисплея или печатающее устройство. Если требуется сохранить эти результаты, то их следует записать в файл, например на магнитный диск. Каждый файл определяется спецификацией, имеющей следующую структуру:

<имя устройства>: <имя файла>.<тип файла>, где имена накопителей на магнитных дисках обозначаются латинскими буквами A, B, C, ...; <имя файла> – имя, записанное по правилам записи имен, которое задается пользователем; <тип файла> – для файлов данных используется стандартное имя DAT.

Различают файлы последовательного и прямого доступа. Файлы последовательного доступа позволяют вводить и выводить данные только подряд. Файлы прямого доступа позволяют ввести или вывести любую запись, указывая его порядковый номер.

Оператор OPEN используется для открытия файлов. Он имеет следующую структуру:

OPEN <спецификация файла> FOR <режим> AS#<номер файла> LEN = <длина>,

где <режим> принимает одно из следующих значений: INPUT – ввод из последовательного файла; OUTRUT – вывод в последовательный файл; APPEND – добавление в конец последовательного файла; <номер файла> – целое число (1, 2, 3, ...); <длина> – длина записи в байтах. Этот оператор используется для открытия последовательного файла.

Другая модификация этого оператора имеет структуру:

OPEN <режим>, # <номер файла>, <спецификация файла>, <длина>,

где <режим> – один из символов (O – вывод в последовательный файл, I – ввод из последовательного файла, R – ввод–вывод в файл прямого доступа), заключенных в кавычки. Остальные параметры те же, что и в предыдущем операторе. Если длина не указана, то считается равной 128 байтам.

Оператор CLOSE используется для закрытия файлов, номера которых указаны в нем.

Ввод и вывод данных в файл последовательного доступа осуществляются с помощью операторов INPUT, PRINT и PRINT USING, в которых указывается после символа # номер файла. Например, операторы

```
10 OPEN "O", #2, "A : STAT. DAT",80
80 PRINT #2, F¤"; "; K1¤; ","; K2¤
```

открывают файл с именем STAT и номером 2 для вывода и выводят в этот файл значения символьных переменных F¤, K1¤, K2¤ (файл расположен на устройстве A).

При работе с файлами прямого доступа используется специальный буфер, который описывается с помощью *оператора* FIELD, имеющего структуру: FIELD #<номер файла>, <длина 1> AS <имя 1>, <длина 2> AS <имя 2>, ..., где <длина> – число байтов памяти, отводимое для хранения значения переменной, <имя> – имя символьной переменной или элемента массива. Суммарная длина всех переменных, указанных в операторе FIELD, должна быть равна длине, указанной в операторе OPEN. Например, операторы

```
10 OPEN "R", #2, "A : STAT", 21
20 FIELD #2, 15 AS F1¤, 3 AS K1¤, 3 AS K2¤
```

открывают файл прямого доступа с именем STAT и номером 2 (расположенный на диске A) для ввода и вывода. Длина записи 21 байт. Запись состоит из символьных переменных F1¤, K1¤, K2¤, имеющих длину 15, 3, 3 байтов соответственно.

При вводе требуемая запись передается из дискового файла в буфер с помощью оператора GET, имеющего структуру: GET #<номер файла>, <номер записи>. Если <номер записи>отсутствует, то вводится очередная запись.

При выводе значения переменных заносятся в буфер с помощью операторов присваивания, имеющих структуру:

LSET <имя> = <символьное выражение>
RSET <имя> = <символьное выражение>

Первый оператор размещает данные в отведенном поле в левых байтах, а второй – в правых.

РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА

Зуев В.М. Лабораторные работы для подготовки термистов. – М.: Высшая школа, 1978.

Гольдин И.И. Основные сведения о сопротивлении материалов, механизмах и деталях машин. – М.: Высшая школа, 1977.

Удачин В.С., Соловьев В.Б. Судовождение и правила плавания на внутренних судоходных путях. – М.: Транспорт, 1983.

Тихомиров Н.А. Теория и устройство судна внутреннего плавания. – М.: Транспорт, 1965.

Гелюта Е.З., Нурмухамедов Ю.К. Горное дело. – М.: Недра, 1965.

Павлов К.Д., Романков П.Г., Носков А.А. Примеры и задачи по курсу процессов и аппаратов химической технологии. – М.: Госхимиздат, 1961.

Кувшинский М.Н., Соболева А.П. Курсовое проектирование по предмету "Процессы и аппараты химической промышленности". – М.: Высшая школа, 1980.

Рубчинский З.М., Соколов С.И., Эглом Е.А., Лынюк Л.С. Электропоезда. – М.: Транспорт, 1983.

Рубчинский З.М., Тастевен Е.Э., Лынюк Л.С., Эглом Е.А. Устройство и работа моторвагонного подвижного состава. – М.: Транспорт, 1969.

ОГЛАВЛЕНИЕ

Предисловие 3

Глава первая. Алгоритмизация задач 5

§ 1. Подготовка задач для решения на ЭВМ 5
§ 2. Примеры составления алгоритмов 10
Задачи для самостоятельного составления алгоритмов решения 21

Глава вторая. Программирование на языке БЕЙСИК 25

§ 3. Простейшие конструкции языка 25
§ 4. Операторы языка 28
§ 5. Примеры составления программ 36
Задачи и упражнения по программированию для самостоятельного решения 44
Ответы и пояснения 52
Приложение 1. Работа с микроЭВМ 88
Приложение 2. Вывод результатов в виде графиков, таблиц, гистограмм 93
Приложение 3 102
Рекомендуемая литература 111

Алексеев Владимир Евтихиевич
Ваулин Анатолий Сергеевич

ЗАДАЧИ И УПРАЖНЕНИЯ ПО ПРОГРАММИРОВАНИЮ

Книга 2.

Заведующая редакцией С.В. Никитина
Редактор В.И. Мучкина
Художники Ю.М. Аратовский и С.Ю. Вериченко
Художественные редакторы В.И. Мешалкин и В.Г. Пасичник
Технические редакторы И.А. Балелина и Н.В. Яшукова
Корректор Р.К. Косинова
Оператор Н.В. Хазраткулова

ИБ № 7895

Изд. № ЭГ-217. Сдано в набор 17.11.88. Подп. в печать 16.01.89.
Формат 60×90^1/$_{16}$. Бум. офс. № 2. Гарнитура Пресс-Роман. Печать офсетная.
Объем 7 усл. печ. л. 7,25 усл. кр.-отт. 6,54 уч.-изд. л.
Тираж 100 000 экз. Зак. №102 Цена 35 коп.
Издательство "Высшая школа", 101430, Москва, ГСП-4, Неглинная ул., д. 29/14.

Набрано на наборно-пишущих машинах издательства. Отпечатано в Ярославском полиграфкомбинате "Союзполиграфпрома" при Государственном комитете СССР по делам издательств, полиграфии и книжной торговли. 150014, Ярославль, ул. Свободы, 97.

35 коп

МАШИНОСТРОЕНИЕ, ПРИБОРОСТРОЕНИЕ И СВЯЗЬ 1

ТЯЖЕЛАЯ ПРОМЫШЛЕННОСТЬ И ТРАНСПОРТ 2

ЛЕГКАЯ ПРОМЫШЛЕННОСТЬ 3

СТРОИТЕЛЬСТВО 4

СЕЛЬСКОЕ ХОЗЯЙСТВО 5