№ 22 (215) июнь 1999

ИНФОРМАТИК А

еженедельное приложение
к газете «ПЕРВОЕ СЕНТЯБРЯ»

Практикум по Турбо Паскалю

И.А. Бабушкина,
Н.А. Бушмелева,
С.М. Окулов,
С.Ю. Черных

ВЫПУСК 1

Содержание*

* Книга выходит в нескольких номерах нашей газеты. Номера страниц указаны у разделов, вошедших в данный номер.

ПРЕДИСЛОВИЕ

Эмоциональное вступление. Книг по языку, который был создан Никлаусом Виртом, издано великое множество, и вы, открыв эту, скажете: опять?! ну сколько можно! Вы и правы, и не правы. Во-первых, язык программирования Паскаль неисчерпаем с точки зрения возможностей развития аналитического ума школьника, так что можем вас огорчить или обрадовать — эта книга не последняя. Во-вторых, подобные книги по большей части представляют собой вариации на тему описания языка. Представим ситуацию: учителю необходимо подготовиться к занятию или школьнику нужно проработать самостоятельно какую-либо тему. Достаточно ли для этих целей описания языка программирования, одного или двух примеров? Профессионал в образовании даст только один ответ: нет. Данная книга потому и называется *конспектами занятий*, что она содержит материал по изучению языка, а не просто очередной вариант его описания. В-третьих, язык программирования Паскаль — это не только прекрасное средство образовательной информатики, но и, как показала история компьютерной науки, основа современных профессиональных систем программирования, например, Delphi. Можно продолжить, но, думается, достаточно.

В книге обобщен более чем семилетний опыт преподавания Паскаля. Резюмируя его, выделим следующие основные положения. Вы не найдете чисто теоретических уроков. Мы полагаем, что информатике они противопоказаны. Превращение уроков информатики по своей структуре в традиционные, такие же, как уроки по истории, математике, при наличии новой дидактической составляющей — КОМПЬЮТЕРА — это переход от авиации к гужевому транспорту. Наша установка очень проста. Если теорию или то, что мы называем теорией, нельзя изложить за 10—15 минут урока и связать с последующей работой за компьютером, самостоятельной работой ученика под руководством учителя, то такую теорию излагать не следует.

Кому адресована эта книга? В первую очередь учителю информатики и, конечно, школьнику и студенту. В каждом разделе содержится материал как для проведения теоретической пятнадцатиминутки, так и для последующей работы с задачами. Школьник и студент найдут в этой книге задачи, решение которых действительно позволит им освоить язык. Количество разобранных задач таково, что читатель не окажется брошенным на произвол судьбы.

Что содержится в этой книге? В первой главе — начала Паскаля. Особенность главы в том, что в "ускоренном режиме" вводятся основные управляющие конструкции языка. Цель — рассмотрение более содержательных задач в дальнейшем. При изучении материала первой главы рекомендуется в полном объеме использовать режим пошагового выполнения программы. Вторая глава начинается с темы "Процедуры и функции", затем рассматриваются файловый тип данных и все основные типы данных Паскаля. Такое построение материала обусловлено следующим: знание первых двух тем закрепляется последующим материалом, обогащает этот материал и позволяет последовательно формировать то, что в информатике называют структурной парадигмой мышления. Третья глава традиционна — "Методы сортировки и поиска данных". В четвертой главе повторно изучается рекурсия. Пятая глава посвящена изучению динамических структур данных: списки; реализация стеков и очередей; деревья.

Что вам должно быть известно и что вы будете знать? Конечно, хорошо бы уметь работать в среде LOGO, но можно начинать и с нуля. В результате вы не только получите знания по системе программирования Паскаль, но и достигнете определенного уровня понимания предмета информатики.

ОТ РЕДАКЦИИ

Уважаемые читатели! В прошедшем учебном году мы опубликовали несколько анкет, результаты анализа которых позволяют делать выводы о популярности материалов различных рубрик нашей газеты. Ваши пожелания учитывались и при планировании летних номеров "Информатики". "Практикум по Турбо Паскалю", который мы начали публиковать в 1997 году, имеет наивысший рейтинг, опережая даже такие "хиты", как "Задачи", "Как это делаю я", "Круглый стол" и другие популярные материалы.

Летом выйдут три спецвыпуска, в которых будут собраны все материалы книги, опубликованные в 1997—1998 гг., а заключительные главы "Практикума" будут опубликованы в осенних номерах.

1. НАЧАЛА ПАСКАЛЯ

1.1. ПЕРВЫЕ ЗАНЯТИЯ

1.1.1. Первое знакомство с системой программирования Турбо Паскаль

Турбо Паскаль появился на рынке программных продуктов в 1984 году и совершил настоящую революцию в программировании. До этих пор при обучении программированию предпочтение чаще всего отдавалось Бейсику — простому, дешевому и легко осваиваемому. Паскаль же был аппаратно зависимым, дорогим и сложным. С появлением Турбо Паскаля положение изменилось. Турбо Паскаль состоит из языка программирования и среды, которая обеспечивает удобную и производительную работу. Изучение Паскаля как языка программирования идет вместе с изучением всей системы Турбо Паскаль.

Язык программирования Паскаль был разработан Н.Виртом в 1968—1970 годах и получил широкое распространение благодаря наглядности программ и легкости изучения. Он послужил основой для разработки других языков программирования (например, Ада, Модула-2).

Первая версия Турбо Паскаля использовалась не очень долго — она появилась в 1983 году, а уже в 1984 году ее заменила вторая версия, которая получила широкое распространение. К осени 1985 года появляется третья версия, еще более удобная в работе.

Четвертая версия (1988 год) представила Турбо Паскаль в новом виде (появилась новая среда, компилятор стал встроенным). Осенью этого же года вышла пятая версия, в которой появился встроенный отладчик. А в 1989 году появилась версия 5.5, позволившая перейти к объектно-ориентированному программированию.

Шестая версия уже обеспечивала многооконный и многофайловый режим работы, использование мыши, применение объектно-ориентированного программирования, обладала встроенным ассемблером и имела другие возможности.

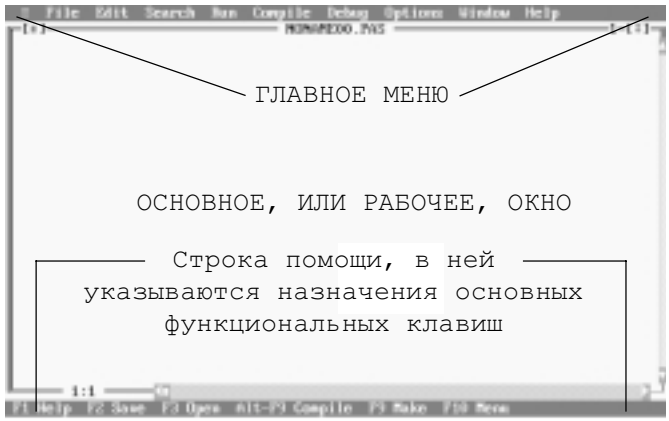
В 1992 году фирма Borland International выпустила два пакета программирования на языке Паскаль — это Borland Pascal 7.0 и Turbo Pascal 7.0.

Пакет Turbo Pascal 7.0 использует новейшие достижения в программировании. Он может использоваться практически на любой машине и относительно дешев. Язык этой версии обладает широкими возможностями, имеет большую библиотеку модулей. Среда программирования позволяет создавать тексты программ, компилировать их, находить и исправлять ошибки, компоновать программы из отдельных частей, использовать модули, отлаживать и выполнять программы.

Первое знакомство

На самом первом занятии учителю необходимо:
- показать, как включать компьютер;
- показать, как правильно вставлять дискету;
- показать, как запускать Турбо Паскаль;
- познакомить с рабочим экраном.

После загрузки системы появляется рабочий экран:



Переход из основного окна в главное меню и наоборот осуществляется при помощи нажатия клавиши F10.

Команды редактора

Примечание. Требуется заранее подготовить карточки с основными командами, необходимыми для работы. Остальные команды учащиеся могут узнать в процессе работы с Турбо Паскалем, используя режим помощи (клавиша F1).

Команды управления движением курсора

→ — перемещение курсора на символ вправо;

← — перемещение курсора на символ влево;

↑ — перемещение курсора на строку вверх;

↓ — перемещение курсора на строку вниз;

Home — перемещение курсора в начало текущей строки;

End — перемещение курсора в конец текущей строки;

Page Up — перемещение курсора на страницу вверх;

Page Down — перемещение курсора на страницу вниз;

Примечание. Страница — это один экран (21 строка).

Ctrl Home — перемещение курсора в левый верхний угол экрана;

Ctrl End — перемещение курсора в левый нижний угол экрана;

Команды вставки и удаления текста

Insert — включение и выключение режима вставки;

Примечание. Если режим вставки включен, то курсор имеет вид горизонтальной мигающей черточки. В режиме вставки набираемый символ вводится в позицию, в которой стоит курсор, а все символы (начиная с символа, стоявшего в позиции курсора), расположенные правее, сдвигаются вправо. Если режим вставки выключен, то набираемый символ заменит тот символ, который находился в позиции курсора, таким образом можно старый текст заменить на новый.

Delete — удаление символа, стоящего в позиции курсора;

Back Space — удаление символа, стоящего слева от курсора;

Примечание. Иногда на этой клавише написано <BS>, а иногда это стрелка ←, расположенная над клавишей ввода Enter).

Ctrl N — вставка пустой строки над строкой, в которой находится курсор;

Ctrl Y — удаление строки, в которой находится курсор.

Задания

1. Набрать в одной строке свою фамилию, имя и отчество. В следующей строке — свой домашний адрес, номер телефона.

Примечание. Учитель должен показать, как переходить в режим ввода русских букв, как набирать заглавные буквы, цифры и знаки препинания.

2. Набрать предложение: "Шла собака по роялю и сложила песню". В этом предложении заменить все буквы "о" на "е", а после каждой буквы "а" вставить букву "с".

Режим помощи

Далее необходимо познакомить с режимом помощи — Help (F1). Надо показать, как входить в режим помощи, переходить от экрана к экрану. Учащиеся подробно знакомятся с этим режимом самостоятельно.

Первая программа

Сначала надо рассказать об общем виде программы.

Программа начинается с заголовка, имеющего следующий вид:

```
Program  <имя программы>;
```

За ним идут разделы описаний, в которых должны быть описаны все идентификаторы (константы, переменные, типы, процедуры, функции, метки), которые будут использованы в программе.

После разделов описаний идет раздел операторов, который начинается со служебного слова **Begin** и заканчивается служебным словом **End**. В этом разделе задаются действия над объектами программы, объявленными в разделе описаний. Операторы в этом разделе отделяются друг от друга точкой с запятой. После последнего слова **End** ставится точка.

Разбор примера

```
Program Example_1;
Var a,b,rez: Integer;
Begin
  Writeln('Введите два числа через пробел');
  Readln(a,b);
  rez:=a*b;
   Writeln('Их произведение равно ', rez);
  Writeln('Нажмите  <Enter>');
  Readln;
End.
```

Пояснения к программе

Имя этой программы Example_1. Заметим, что в имени программы не должно быть пробелов, оно должно начинаться с буквы, состоять только из латинских букв, цифр и некоторых специальных символов (в нашем примере использован символ "подчеркивание"). Из разделов описаний имеется лишь один — раздел описания переменных. Он начинается со служебного слова **Var**, после которого идет последовательность объявлений переменных, разделенных точкой с запятой. В каждом объявлении перечисляются через запятую имена переменных одного типа, после чего ставится двоеточие и указывается тип переменных. В нашем примере описаны три переменные: все они (a, b и rez) имеют целый тип (Integer), то есть значения переменных этого типа — целые числа.

После раздела описаний переменных идет раздел операторов, начинающийся со служебного слова **Begin**, после которого расположены операторы языка. Первый оператор — это Writeln('текст') — записать (вывести) на экран текст, заключенный между апострофами, ln добавляется в конце этого оператора для того, чтобы курсор автоматически переходил в начало следующей строки.

Следующий оператор — Readln(a,b) — читать данные с клавиатуры. В данном случае необходимо ввести два целых числа через пробел, тогда переменной a

присваивается значение, равное первому введенному числу, а переменной b присваивается значение, равное второму введенному числу. Например, вы ввели числа 12 и 45, тогда a=12, а b=45. В конце этого оператора также можно ставить ln.

После этих двух операторов стоит оператор присваивания: rez:=a*b; (:= — это знак присваивания в языке Паскаль). При выполнении этого оператора переменная rez получит значение, равное произведению числа a на число b (*рис.* 1). Так как в результате умножения двух целых чисел получается целое число, то переменная rez описана как целая (Integer).

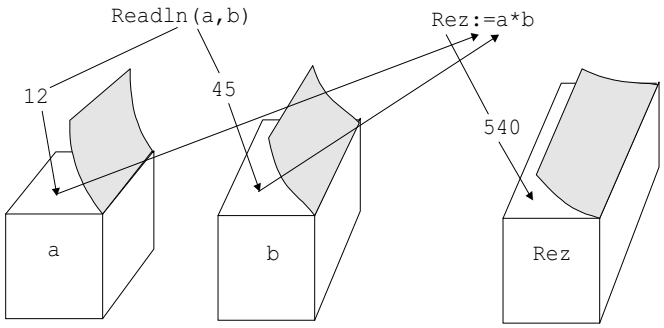


Рис. 1

Следующий оператор — это снова оператор вывода Writeln('текст',rez) — он выведет на экран текст, заключенный между апострофами, а за ним значение переменной rez. Затем следующий оператор — Writeln — выведет на экран сообщение: Нажмите Enter, а оператор Readln будет ожидать нажатия указанной клавиши. В конце раздела операторов стоит служебное слово **End**, после которого ставится точка.

Запуск программы

Для того чтобы запустить программу, необходимо выйти в главное меню (например, посредством клавиши F10) и выбрать режим **Run**. После запуска программы на экране появится сообщение:

Введите два целых числа через пробел

Курсор окажется в следующей строке. Затем надо ввести два целых числа через пробел и нажать клавишу Enter. Появится сообщение:

Их произведение равно...

Вместо точек будет напечатано значение переменной rez, то есть число, равное произведению первого введенного числа на второе. Это сообщение останется на экране до тех пор, пока не будет нажата клавиша Enter.

Задания

1. Измените программу таким образом, чтобы в ней вычислялась сумма двух чисел.
2. Измените программу таким образом, чтобы в ней вычислялась сумма четырех чисел.
3. Напишите программу для вычисления значения выражения (a+(d–12)*3)*(c–5*k); значения переменных a,d,c и k вводятся с клавиатуры.
4. Выведите на экран в одной строке свою фамилию, имя и отчество, в следующей строке — дату рождения.

Сохранение программы

Для того чтобы сохранить программу, необходимо выйти в главное меню и выбрать пункт **File**. Затем в появившемся вертикальном меню надо выбрать пункт **Save as...** Появится окно, в котором можно ввести имя файла. Например, a:\prim1_1.pas; здесь a: — это название диска, "\" — каталог (корневой), prim1_1 — имя файла (оно может содержать не более 8 символов), pas — расширение, указывающее, что файл содержит программу, написанную на языке Паскаль.

Примечание. Следует сообщить учащимся список символов, которые нельзя употреблять в именах файлов: * = + [ ] \ |;:,. < > /?. Также не следует использовать в именах файлов символ пробела и буквы русского алфавита.

После того как имя файла набрано, нажмите клавишу Enter.

Примечание. Для быстрого сохранения файла можно воспользоваться командами **Save** или **Save all** меню **File**.

Выход из системы программирования Турбо Паскаль

Для того чтобы закончить работу, необходимо выбрать пункт **Quit** в меню **File** или просто набрать комбинацию клавиш Alt X.

1.1.2. Простейшие линейные программы

Вопросы для повторения

1. Назовите команды удаления, вставки символов и передвижения курсора.
2. С чего начинается программа?
3. Как описываются переменные?
4. С чего начинается основная программа?
5. Как записывается оператор вывода?
6. Как записывается оператор ввода?
7. Как записывается оператор присваивания?
8. Чем заканчивается программа?
9. Как сохранить программу на диске?

Решение задач

1. Найти периметр:

a) прямоугольника, ширина и длина вводятся с клавиатуры;

b) треугольника, длины всех сторон вводятся с клавиатуры;

c) произвольного четырехугольника, длины всех сторон вводятся с клавиатуры.

2. Вычислить значение выражений:

a) $y=15x^2+8x-9$;

b) `a=(b+c)*d-k`.

Арифметический квадрат. Абсолютная величина

Функция `SQR(x)` возвращает квадрат значения аргумента, то есть $SQR(x)=x^2=x*x$.

Примеры

$sqr(4)=4^2=16$;

при x=13, $sqr(x)=sqr(13)=13^2=169$;

при d=2, e=5, $sqr(d+e)=sqr(2+5)=sqr(7)=49$;

при x=3, $sqr(sqr(x))=sqr(sqr(3))=sqr(9)=81$.

Функция `ABS(x)` возвращает абсолютную величину значения аргумента.

$$ABS(x)=\begin{cases} x, & \text{при } x\ge 0;\\ -x, & \text{при } x<0. \end{cases}$$

Примеры

```
abs(12)=12;
abs(-12)=12;
при x=3, y=-5;
abs(x+y)=abs(3+(-5))=abs(-2)=2;
abs(x)+abs(y)=abs(3)+abs(-5)=3+5=8.
```

Примечание. В Паскале большие и маленькие буквы в именах (переменных, функциях и пр.) не различаются. Не различаются они и при записи служебных слов.

Решение задач

1. Вычислить рациональным способом, то есть за минимальное количество операций:

a) $y=x^5$ ($y=(x^2)^2*x$, 3 операции);

b) $y=x^6$ ($y=(x^3)^2=(x^2*x)^2$, 3 операции);

c) $y=x^8$ ($y=((x^2)^2)^2$, 3 операции).

2. Найти значение выражений:

a) $y=|x|$, при x=-3, x=3;

b) $a=|x|$, при x=2, x=-2;

c) $z=|x-2|+3x^8$, при x=-2, x=1;

d) $a=6b^2+|b-3|^3-15$, при b=9, b=-3.

3. Написать программу вычисления значения выражений:

a) $y=(3x^3+18x^2)*x+12x^2-5$;

b) `a=(d+c+b)*e-5k-l`;

c) $d=3c^3+|c^2-4c+7|^3-5c$;

d) $c=|x+4|-|x^2-3x+6|$.

4. Даны значения переменных x и y: x=14, y=3. Какими будут значения этих переменных после выполнения последовательности действий:

a) `x:=y; y:=x;`

b) `d:=x+1; x:=y; y:=d.`

5. Поменять местами значения переменных x и y:

a) с использованием дополнительной переменной (`t:=x; x:=y; y:=t;`);

b) без использования дополнительной переменной (`x:=x-y; y:=x+y; x:=y-x;`).

1.1.3. Целый и логический типы данных. Условный оператор

Повторение

1. Какую функцию можно использовать при записи выражения $y=x^2+3x-7$ на языке Паскаль?

2. Записать на языке Паскаль следующие выражения:

a) $y=5x^5-10x+2$;

b) $z=14x^4-5x^3+11x-17$.

3. Какая функция используется при записи выражения $y=|x-4|+12$ на языке Паскаль?

4. Записать на языке Паскаль следующие выражения:

a) $a=x^3+|x^2-13x+5|-11$;

b) $s=|3x^4+12x^3-4x+7|-13$.

Целый тип данных

Переменные целого типа описываются посредством идентификатора `Integer`. Они могут принимать значения в диапазоне от —32 768 до 32 767. К данным целого типа можно применять операции "+" — сложение, "—" — вычитание, "*" — умножение и некоторые другие.

Так как в результате деления одного целого числа на другое не всегда получается целое число, то имеются операции:

`div` — целая часть от деления;

`mod` — остаток от деления.

Примеры

```
19 div 4=4;
12 div 4=3;
-21 div 4=-5;
-7 div (-4)=1;
19 mod 4=3;
12 mod 4=0;
-21 mod 4=-1;
-7 mod (-4)=-3.
```

Примечание. Переменной целого типа присваивать значение, получаемое в результате выполнения обычной операции деления "/", ***нельзя***, так как при делении одного целого числа на другое целое число результат не всегда является целым числом.

Решение задач

1. Найти целую часть и остаток от деления целого числа `a` на целое число `b`.

2. Найти сумму цифр заданного трехзначного числа.

Логический тип данных

Переменные логического типа описываются посредством идентификатора `Boolean`.

Они могут принимать только два значения — `FALSE` (*ложь*) и `TRUE` (*истина*).

Переменные логического типа обычно получают значения в результате выполнения операций сравнения (отношения): "<" (меньше), ">" (больше), "≤" (меньше или равно), "≥" (больше или равно), "<>" (не равно), "=" (равно). Результат операции отношения равен `TRUE`, если отношение удовлетворяется для значений входящих в него операндов, и `FALSE` в противном случае.

В языке Турбо Паскаль имеются логические операции, применяемые к переменным логического типа. Обозначения и результаты этих операций приведены в табл. 1.

Таблица 1

Значения операндов		Результат операции			
X	Y	not X	X and Y	X or Y	X xor Y
false	false	true	false	false	false
false	true	true	false	true	true
true	false	false	false	true	true
true	true	false	true	true	false

Логические операции, операции отношения и арифметические операции часто встречаются в одном выражении. При этом отношения, стоящие слева и справа от знака логической операции, должны быть заключены в скобки, поскольку логические операции имеют более высокий приоритет. Вообще принят следующий приоритет операций:

- `not`;
- `and, *, div, mod`;
- `or, xor, +, –`;
- операции отношения.

Кроме того, порядок выполнения операций определяется скобками. Например, в логическом выражении `A or B and not (A or B)` сначала выполняется заключенная в скобки операция `or`, а затем операции `not, and, or`.

В языке Паскаль нет возможности ввода логических данных с помощью оператора `Read`. Однако предусмотрен вывод значений переменных логического типа с помощью оператора `Write`. При вводе для идентификаторов `FALSE` и `TRUE` отводится по 6 позиций, а сами идентификаторы прижимаются к правому краю поля вывода.

Задания

Вычислить значения выражений:

a) `(a>5) and (b>5) and (a<20) and (b<30);`

b) `not(a<15) or not(b<30);`

c) `c or d and (b=20)`

при `a=10, b=20, c=true, d=false`.

Условный оператор

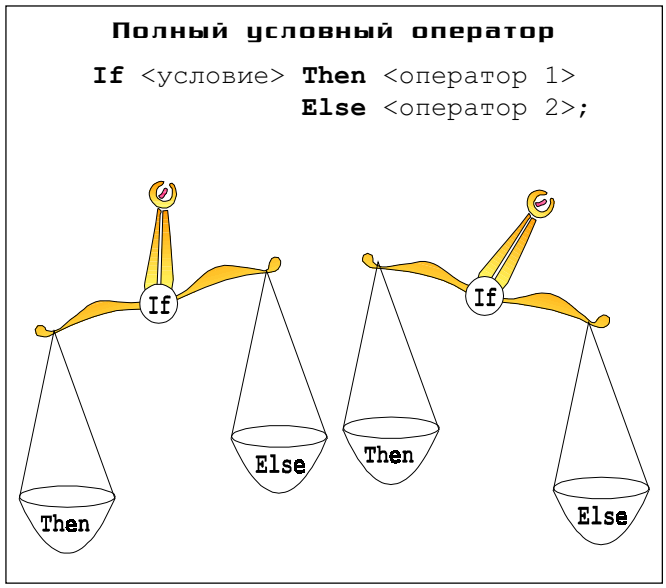


Рис. 2

Выполнение условного оператора начинается с вычисления значения логического выражения, записанного в условии. Если условие истинно, то выполняется <оператор1>, в противном случае — <оператор 2> (см. *рис.* 2). Если в качестве оператора должна выполниться серия операторов, то они заключаются в операторные скобки **`Begin-End`**.

Разбор условного оператора можно выполнить на следующем простом примере.

Пример

Вывести на экран большее из двух данных чисел.

```
Program Example_2;
Var x,y: Integer;
Begin
  Writeln('введите 2 числа');
  {вводим два целых числа через пробел }
  Readln(x,y);
  If x>y Then Writeln(x)
  {если (If) x больше y, то (Then)  выводим  x,}
         Else Writeln(y);
         {иначе (Else) выводим y}
  Readln;
End.
```

Введем два числа — 5 и 7. Переменная x получит значение 5, а переменная y — значение 7 (x=5, y=7). Условие x>y не выполняется, так как 5 не больше 7. Управление передается на оператор, стоящий после Else, то есть Writeln(y), а следовательно, на экране появится 7.

Примечание. Обратите внимание учащихся на то, что перед служебным словом Else разделитель — точка с запятой — не ставится.

Неполный условный оператор

If <условие> **Then** <оператор>;

Ветвь **Else** может отсутствовать, если в случае невыполнения условия ничего делать не надо. Например, рассмотрим следующую задачу: если значение переменной x меньше 0, то поменять его на противоположное. Задача решается с помощью такого условного оператора:

```
If x<0 Then x:=-x;
```

Если в условном операторе имеется ветка Else, он называется полным, в противном случае — неполным.

Пример

Написать программу, проверяющую, принадлежит ли число, введенное с клавиатуры, интервалу (0;5).

Решение

Обозначим через x число, вводимое с клавиатуры пользователем (это переменная целого типа). Принадлежность числа x интервалу (0;5) определяется следующей системой неравенств: $\begin{cases} x>0 \\ x<5 \end{cases}$

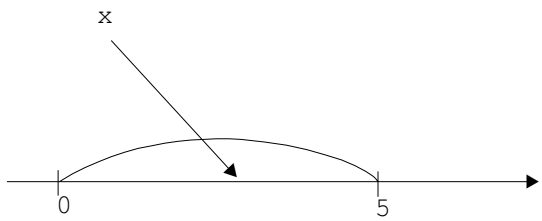


Рис. 3

То есть x принадлежит заданному интервалу лишь в том случае (см. *рис. 3*), если одновременно выполняются оба условия: (x>0) и (x<5).

```
Program Example_3;
Var x: Integer;
Begin
  Writeln('Введите число x');
  Readln(x);
  If (x>0)and(x<5)
    Then Writeln(x,' принадлежит (0,5)')
   Else Writeln(x,' не принадлежит (0,5)');
End.
```

Вложенные условные операторы

При решении задач часто приходится рассматривать не два, а большее количество вариантов. Это можно реализовать, используя несколько условных операторов. В этом случае после служебных слов Then и Else записывается новый условный оператор.

Пример

Даны целые числа a,b,c. Если a≤b≤c, то все числа заменить их квадратами, если a>b>c, то каждое число заменить наименьшим из них, в противном случае сменить знак каждого числа.

Решение

Условие задачи перепишем следующим образом:
a:=a^2, b:=b^2, c:=c^2, если a≤b≤c
a:=c, b:=c, если a>b>c
a:=-a, b:=-b, c:=-c — в остальных случаях.

```
Program Example_4;
Var a,b,c: Integer;
Begin
  Writeln('Введите числа a, b, c');
  Readln(a,b,c);
  If (a<=b) and (b<=c)
    Then Begin
     a:=sqr(a); b:=sqr(b); c:=sqr(c)
         End
    Else If (a>b) and (b>c)
          Then Begin a:=c; b:=c End
           Else Begin a:=-a; b:=-b; c:=-c End;
  Writeln(a:3,b:3,c:3);
  Readln
End.
```

Примечание. Если вложенными условными операторами являются неполные условные операторы (как, например, в задаче, рассмотренной выше), то могут возникать неясности, связанные с установлением границ условных операторов. В таких случаях служебное слово Else относится к ближайшему If.

Задание

В приведенной выше программе изменим условный оператор следующим образом:

```
If (a<=b) and (b<=c) Then
  Begin
   a:=sqr(a); b:=sqr(b); c:=sqr(c)
   If (a>b) and (b>c) Then
     Begin c:=a; b:=a End
   Else Begin a:=-a; b:=-b; c:=-c End;
  End;
```

Как изменится выполнение данной программы? Сформулируйте условие задачи, которую теперь решает данная программа.

Решение задач

1. Определить значение логического выражения:
`(–3>=5) or not(7<9) and(0<3)`

2. Имеется условный оператор:

```
If d<>10 Then Writeln('ура!')
Else Writeln('плохо...');
```

Можно ли заменить его следующими операторами:

```
If d=10 Then Writeln('ура!')
Else Writeln('плохо...');
```

```
If not(d=10) Then Writeln('ура!') Else
Writeln('плохо...');
```

```
If not(d=10) Then Writeln('плохо...') Else
Writeln('ура!');
```

```
If not(d<>10) Then Writeln('плохо...') Else
Writeln('ура!');
```

3. Какими будут значения переменных `j, k` после выполнения условного оператора:

```
If j>k Then j:=k–2 Else dec(k,2);
```

— если исходные значения переменных равны:

a) `j=3, k=5;`
b) `j=3, k=3;`
c) `j=3, k=2.`

В результате выполнения оператора `dec(k,2)` значение переменной `k` уменьшается на 2.

4. Запишите условный оператор, в котором значение переменной `c` вычисляется по формуле `a+b`, если `a` — нечетное и `a*b`, если `a` — четное.

5. Вычислить значение функции:

$$\begin{cases} x^2+5, & \text{при } x>3; \\ x-8, & \text{при } x\le 3. \end{cases}$$

6. Найти наибольшее из трех данных чисел.

7. Вывести на экран номер четверти, которой принадлежит точка с координатами `(x,y)`, при условии, что `(x<>0)` и `(y<>0)`.

8. Вычислить значения функции:

$$\begin{cases} x-12, & \text{при } x>0 \\ 5, & \text{при } x=0 \\ x^2, & \text{при } x<0 \end{cases}$$

9. Даны три различных целых числа, найти среднее из них. Средним назовем число, которое больше наименьшего из данных чисел, но меньше наибольшего.

10. Написать фрагмент программы для подсчета суммы только положительных из трех данных чисел.

11. Даны три числа. Написать фрагмент программы для подсчета количества чисел, равных нулю.

12. После выполнения операторов

```
a:=0;
If a<>0 Then; a:=2;
```

значение переменной `a` равно двум. Объясните почему.

13. Используя составной оператор, упростите следующий фрагмент программы:

```
If a>b Then c:=1;
If a>b Then d:=2;
If a<=b Then c:=3;
If a<=b Then d:=4;
```

14. Каким будет значение переменной `a` после выполнения операторов:

```
a:=3;
If a<4 Then
        Begin Inc(a,2); Inc(a,3); End.
```

В результате выполнения оператора `Inc(a,3)` значение переменной `a` увеличивается на 3.

15. Составьте программу нахождения произведения двух наибольших из трех введенных с клавиатуры чисел.

16. Если целое число `M` делится нацело на целое число `N`, то вывести на экран частное от деления, в противном случае вывести сообщение: `"M на N нацело не делится"`.

17. Найти количество положительных (отрицательных) чисел среди четырех целых чисел `A, B, C` и `D`.

18. Чему равны значения переменных `a` и `b` после выполнения последовательности действий:

a)
```
a:= 15 Div (16 Mod 7);
b:= 34 Mod a * 5 – 29 Mod 5 * 2;
```
b)
```
a:= 4 * 5 Div 3 Mod 2;
b:= 4 * 5 Div ( 3 Mod 2);
```
c)
```
a:= a*b;b:= b*b.
```

19. Составьте программу, которая определяет вид треугольника по длинам его сторон (если данные длины позволяют построить треугольник).

20. Составьте программу, которая уменьшает первое введенное число в пять раз, если оно больше второго введенного числа по абсолютной величине.

21. Составьте программу для вычисления выражения

a) `max(x+y+z,xyz)+3;`
b) $\min(x^2+y^2, y^2+z^2)-4$.

Значения переменных `x,y,z` вводятся с клавиатуры.

22. Составьте программу, в которой из трех введенных с клавиатуры чисел возводятся в квадрат положительные, а отрицательные остаются без изменения.

1.1.4. Целый тип данных. Цикл с параметром

Повторение

1. Какие операции можно применять к переменным целого типа?

2. Можно ли присваивать результат операции деления (`/`) переменной целого типа? Почему?

3. Напишите тестовые программы и изучите работу следующих операторов:

a) `Write('AB':3);Writeln(5*2:3);`
b) `Writeln(5:6,8:5);Writeln('КОНЕЦ');`
c) `Writeln('РЕЗУЛЬТАТ ',5*3,3).`

Обратите внимание на то, сколько позиций отводится для вывода каждого из параметров оператора `Writeln`.

4. Что такое `If, Then, Else`?

5. Как выглядит полный условный оператор? Как он работает?

6. Как выглядит неполный условный оператор? Как он работает?

7. Записать на языке Паскаль следующие действия:

a) если число является четным, то вывести "ДА", иначе вывести "НЕТ";

b) если число делится на 5, то вывести целую часть от деления.

8. Записать на языке Паскаль следующую формулу:

$$x=\begin{cases} -5, & \text{если } x<-5;\\ x, & \text{если } -5\le x\le 0;\\ 2x, & \text{если } 0<x\le 3;\\ 6, & \text{если } x>3. \end{cases}$$

Можно ли сделать это без вложенных операторов условия? Почему?

Цикл с параметром

Оператор цикла с параметром

```
For <параметр>:= A To B Do
   <тело цикла>;
For <параметр>:=A Downto B Do
   <тело цикла>;
```

где A — начальное значение параметра,
B — конечное значение параметра.

Оператор цикла с параметром применяют тогда, когда заранее известно число повторений одной и той же последовательности операторов.

Начальное и конечное значения параметра цикла могут быть представлены константами, переменными или арифметическими выражениями.

Рассмотрим, как выполняется оператор цикла с параметром вида

```
For  <параметр>:=A To B Do <тело цикла>
```

Сначала вычисляются значения выражений A и B. Если A≤B, то <параметр> последовательно принимает значения, равные A, A+1, ..., B−1, B и для каждого из этих значений выполняется <тело цикла>. Если A>B, то <тело цикла> не выполняется ни разу.

Оператор цикла с параметром

```
For <параметр>:=A Downto B Do <тело цикла>
```

выполняется аналогичным образом, но значение <параметра> изменяется с шагом, равным −1.

Если <тело цикла> состоит из нескольких операторов, то операторы тела цикла заключаются в операторные скобки **Begin-End.**

Пример

Составить программу вычисления значения выражения $y=((\ldots(20^2-19^2)^2-18^2)^2-\ldots-1^2)^2$.

Решение

В данном случае целесообразно организовать цикл с параметром, изменяющимся от 19 до 1, то есть шаг изменения параметра равен −1.

Обозначим через y очередное значение квадрата числа, а через n — параметр цикла.

```
Program Example_5;
Var  y, n: Integer;
Begin
  y:=sqr(20);
   For n:=19 Downto 1 Do y:=sqr(y-sqr(n));
  Writeln('Значение выражения равно');
  Writeln(y);
End.
```

Пример

Из чисел от 10 до 99 вывести те, сумма цифр которых равна s (0<n≤18).

Вопросы для обсуждения

1. Каким образом можно выделить последнюю (младшую) цифру числа?

2. Каким действием можно выделить первую (старшую) цифру числа?

Обозначим через k очередное число, p1 — старшую цифру числа k, p2 — младшую цифру числа k, s — сумму цифр числа k. Число k будем печатать только в том случае, когда сумма p1 и p2 будет равна s.

```
Program Example_6;
Var k,n,p1,p2,s:Integer;
Begin
  Writeln('введите целое число ');
  Readln(n); {вводим целое число}
  For k:=10 To 99 Do {для(For) k от 10
                      до(To) 99 делать(Do)}
  Begin
   p1:=k div 10; {выделяем старшую цифру}
   p2:=k mod 10; {выделяем младшую цифру}
   s:=p1+p2;      {находим сумму цифр}
   If s=n Then Writeln(k);
   {если сумма равна n, то выводим k}
  End;
  Readln;
End.
```

Пример

Найти все двузначные числа, которые делятся на n или содержат цифру n.

Решение

Если двузначное число удовлетворяет условию задачи, то для него выполняется хотя бы одно из трех условий: первая цифра равна n (p1=n), или вторая цифра равна n (p2=n), или само число делится на n (k mod n = 0).

Какую логическую операцию необходимо использовать для объединения этих простых условий?

Решение задач

1. Сколько раз будет выполнено тело цикла в следующих фрагментах программ:

```
a)  For k:=-1 To 1 Do ...
b)  For k:=10 To 20 Do ...
c)  For k:=20 To 10 Do ...
d)  k:=5; r:=15;
    For i:=k+1 To r-1 Do ...
e)  k:=5;r:=15;
    For i:=0 To k*r Do ...
f)  k:=r;
    For i:=k To r Do ...
```

2. Определить значение переменной s после выполнения следующих операторов:

```
s:=0; n:=10;
For i:=2 To n Do s:=s+100 div i;
```

3. Составить программу возведения натурального числа в квадрат, используя следующую закономерность:

$1^2 = 1$
$2^2 = 1+3$
$3^2 = 1+3+5$
$4^2 = 1+3+5+7$
.....
$n^2 = 1+3+5+7+9+...+2n-1$

4. Определить количество трехзначных натуральных чисел, сумма цифр которых равна заданному числу n.

5. Составить программу вычисления суммы кубов чисел от 25 до 125.

6. Среди двузначных чисел найти те, сумма квадратов цифр которых делится на 13.

7. Написать программу поиска двузначных чисел, обладающих следующим свойством: если к сумме цифр числа прибавить квадрат этой суммы, то получится снова данное число.

8. Квадраты некоторых трехзначных чисел оканчиваются тремя цифрами, которые как раз и составляют исходные числа. Написать программу поиска таких чисел.

9. Написать программу поиска четырехзначного числа, которое при делении на 133 дает в остатке 125, а при делении на 134 дает в остатке 111.

10. Найти сумму положительных нечетных чисел, меньших 100.

11. Найти сумму целых положительных чисел из промежутка от A до B, кратных 4 (значения переменных A и B вводятся с клавиатуры).

12. Найти сумму целых положительных чисел, больших 20, меньших 100, кратных 3 и заканчивающихся на 2, 4 или 8.

1.1.5. Работа с окнами. Метод пошагового выполнения программ

Повторение

1. Как сохранить программу?

2. Как открыть уже имеющийся файл с программой, написанной на языке Паскаль?

3. Откройте программы с именами Example_3.pas, Example_4.pas, Example_5.pas, Example_6.pas.

4. Как располагаются программы на экране?

Работа с окнами

Итак, каждый файл располагается в своем окне. Можно открывать любое количество окон, но активным является только одно окно, в котором находится курсор. Активное окно находится над всеми другими окнами. Чтобы сделать окно с номером N активным, необходимо нажать комбинацию клавиш Alt и номер окна (клавишу с цифрой n).

Все команды для работы с окнами находятся в пункте **Windows** главного меню. При открытии нескольких файлов они загружаются в окна, которые накладываются одно на другое.

Чтобы посмотреть список открытых окон, можно воспользоваться комбинацией клавиш Alt 0 или командой **List** меню **Windows**.

Для закрытия окна можно воспользоваться командой **Close** меню **Windows** (или комбинацией клавиш Alt F3). Для закрытия всех окон нужно выбрать команду **Close all** меню **Windows**.

Задания

1. Вывести на экран список открытых окон.

2. Перейти в окно с файлом Example_3.pas.

3. Перейти в окно с номером 3.

4. Закрыть окна с номерами 1 и 2.

При выборе пункта **Size/Move** меню **Windows** ограничивающие линии окна меняют цвет. В это время с помощью клавиш управления курсором и клавиши Shift можно изменять размеры окна, без нажатия клавиши Shift можно изменять положение окна на экране. После выбора нужного размера и положения нажмите клавишу Enter. Выбор команды **Zoom** увеличивает размеры активного окна до максимального.

При составлении программ нередко возникает ситуация, когда программа работает не так, как предполагает программист. В этом случае требуется проследить выполнение программы по шагам. В среде Турбо Паскаль есть такая возможность. Для выполнения программы в пошаговом режиме требуется выполнить команду **Step Over** меню **Run** или нажать функциональную клавишу F8.

Примечание. Прежде чем проводить пошаговую отладку программы, необходимо убедиться, что опция **Options/Debugger/Integrated** активизирована.

Чтобы проследить за выполнением программы, нужно знать, как изменяются значения переменных. Активизируем окно **Watches**, выполнив команду **Watch** меню **Debug**. Чтобы ввести в окно **Watches** какую-либо переменную, воспользуемся комбинацией клавиш Ctrl F7 (или выполним команду **Add Watch** меню

Debug), после чего откроется диалоговое окно **Add Watch**, в которое необходимо ввести имя переменной.

Примечание. Чтобы облегчить процесс ввода переменных в окно **Watches**, можно перед нажатием комбинации клавиш Ctrl F7 установить курсор имени переменной, которую вы хотите ввести.

Для удаления переменной из окна **Watches** войдите в окно **Watches**, с помощью клавиш управления курсором выберите нужную переменную и нажмите клавишу Delete.

Задание

- Загрузите файл `Example_6.pas`.
- Откройте окно **Watches** и поместите в него переменные `p1, p2, k, s`.
- Проследите работу программы в пошаговом режиме и составьте следующую таблицу для значений `k` от `10` до `15`:

k	p1	p2	s

Примечание. Для удобства следует уменьшить размеры окна **Watch** и удобно расположить его на экране.

Иногда в процессе отладки возникает необходимость отладить в пошаговом режиме не всю программу, а лишь ее часть. В этом случае можно воспользоваться командой **Go To Cursor** меню **Run** (или просто нажать клавишу F4), предварительно установив курсор на так называемую строку останова (строка, до которой хотим выполнить программу). Программа будет выполнена до этой строки, а далее вы сможете выполнять программу в пошаговом режиме либо снова воспользоваться командой **Go To Cursor** и выполнить программу до вновь выбранной строки останова.

Кроме того, можно установить в некоторой строке так называемую точку останова (можно установить несколько точек останова). Программа будет выполняться до тех пор, пока не достигнет точки останова. Чтобы установить точку останова, переместите курсор в нужную строку и нажмите комбинацию клавиш Ctrl F8 (или выполните команду **Add breakpoint** меню **Debug**), соответствующая строка будет отмечена подсветкой. После повторного нажатия Ctrl F8 точка останова снимается.

Какими свойствами обладает точка останова?

Активизируем команду **Breakpoint** меню **Debug**. Появится таблица с указанием имени файла, номера строки, в которой находится точка останова, здесь же можно указать условие, выполнение которого будет приводить к прерыванию работы программы, или количество проходов контрольной точки (после выполнения которых произойдет останов).

Установим точку останова на строке

```
s:=p1+p2;{находим сумму цифр}
```

и нажмем клавишу **Edit**. Далее в строке **Condition** введем выражение `s=n`. Запустим программу. При каких значениях `p1` и `p2` программа приостановила свою работу?

Верните все в исходное положение и измените значение в строке **Pass count** на значение, равное `35`. Запустите программу. При каких значениях `p1` и `p2` программа приостановила работу сейчас?

Запустите программу, установив одновременно и условие, и число проходов контрольной точки, после которой должен произойти останов. А теперь при каких значениях `p1` и `p2` программа приостановила свою работу? Как вы думаете, почему это произошло?

Решение задач

Примечание. При решении задач следует использовать метод пошаговой отладки программы.

1. Составить программу возведения данного натурального числа `a` в степень `n`. Для различных `a` экспериментально определите максимальное значение `n`.

2. Даны натуральные числа `a, b`. Вычислить произведение `a*b`, используя лишь операцию сложения.

3. Пусть `n` — натуральное число и пусть `n!!` обозначает произведение `1*3*5*...*n` для нечетного `n` и `2*4*...*n` для четного `n`. Для заданного натурального `n` вычислить `n!!` и $(-1)^{n+1}*n!!$

4. Даны натуральные числа $n, a_1, a_2, \ldots, a_n$.

a) Определить количество членов последовательности $a_1, a_2, \ldots, a_n$, имеющих четные порядковые номера и являющихся нечетными числами.

b) Получить сумму тех чисел данной последовательности, которые удовлетворяют условию $|a_i|<i^2$.

c) Верно ли, что в последовательности больше отрицательных членов, чем положительных?

d) Найти $\min(a_2,a_4,\ldots) + \max(a_1,a_3,\ldots)$.

5. Даны натуральные $n, b_0, b_1, \ldots, b_n$. Вычислить $f(b_0)+f(b_1)+\ldots+f(b_n)$,

$$f(x)=\begin{cases} x^2, \text{ если } x \text{ кратно } 3; \\ x, \text{ если } x \text{ при делениии на } 3 \text{ дает остаток } 1; \\ \left[\frac{x}{3}\right] \text{ в остальных случаях.} \end{cases}$$

6. Дано натуральное число `n`. Получить все его натуральные делители.

7. Даны натуральные числа `m, n`. Получить все кратные им числа, меньшие `m*n`.

1.1.6. Решение задач с использованием цикла с параметром

Повторение

1. Как записывается оператор цикла с параметром? Как он выполняется?

2. Допустим, что тело цикла должно содержать два оператора. Какие из приведенных операторов правильные, какие нет и почему?

```
a) For i:=12 To 15 Do s:=s+i;
b) For a:=30 To 20 Do
     If a Mod 3=0 Then d:=d+1;
c) For x:=1 To 20 Do s:=s+x;
     If (x Mod 2=0) or (x Mod 3=0)
     Then d:=d+1;
```

3. Задать с помощью условного оператора (используя при необходимости вложенные условные операторы) следующие действия:

a) меньшее из двух данных чисел (a и b) заменить на 0, а в случае их равенства заменить оба;

b) большее из трех данных чисел (a,b и c) уменьшить на 5.

Решение задач

1. В трехзначном числе зачеркнули первую цифру слева. Когда полученное двузначное число умножили на 7, то получили исходное число. Найти исходное число. *Ответ*: 350.

2. Сумма цифр трехзначного числа кратна 7, само число также делится на 7. Найти все такие числа.

3. Дано натуральное число n (n≤9999). Определить, является ли оно палиндромом (перевертышем), считая, что оно записано четырьмя цифрами. Например, палиндромами являются числа: 2222, 6116, 0440.

Вопросы для обсуждения

1. Дано число n. Каким образом можно построить перевертыш данного числа?

2. Сколько переменных необходимо для решения данной задачи? Объясните назначение каждой переменной.

Решение

Обозначим через n вводимое число, m — дубликат числа n, a — перевертыш числа n, i — переменная цикла для создания перевертыша.

```
Program Example_7;
Var n,m,a,i:Integer;
Begin
  Writeln('введите целое число,
           не большее 9999');
  Readln(n); {вводим целое число}
  m:=n; a:=0;
     {создание перевертыша}
  For i:=1 To 4 Do    {так как число
                    четырехзначное }
  Begin
    a:=a*10+m Mod 10; m:=m div 10;
  End;
  If a=n Then Writeln('ДА!')
  Else Writeln('НЕТ!')';
  {если перевертыш равен данному числу,
   то выводим "ДА", иначе - "НЕТ"}
  Readln;
End.
```

Рассмотрим выполнение этой программы в пошаговом режиме для числа 3994.

Трассировка (пошаговое выполнение) примера:

i	n	m	a
-	3994	3994	0
1	3994	399	0*10+3994 mod 10=0+4=4
2	3994	39	4*10+399 mod 10=40+9=49
3	3994	3	49*10+39 mod 10=490+9=499
4	3994	0	499*10+3 mod 10=4990+3=4993

Так как значение переменной a не равно значению переменной n, то на экране появится НЕТ!

3. Даны натуральные числа n, k (n, k≤9999). Из чисел от n до k выбрать те, запись которых содержит ровно три одинаковые цифры. Например, числа 6766, 5444, 0006, 0060 содержат ровно три одинаковые цифры.

Решение

Если данное число содержит ровно три одинаковые цифры, то только одна из цифр отличается от остальных, то есть возможны четыре случая (в строках и столбцах указаны позиции в числе).

	1	2	3	4	
1	x	x	x		— первое условие
2	x	x		x	— второе условие
3	x		x	x	— третье условие
4		x	x	x	— четвертое условие

В первом случае отличается последняя цифра, во втором — третья, в третьем — вторая, а в четвертом — первая. Для каждого числа, удовлетворяющего условию задачи, выполняется только одно из условий.

Фрагмент решения

```
Begin
  Writeln(' Введите два числа,
            не больших 9999');
  Readln(n, k);
  For i:=n To k Do
```

```
    Begin
     m:=i;
     {Выделяем цифры: a1 — первая,
      a2 — вторая,a3 — третья,
      a4 — четвертая}
     a4:=m mod 10; m:=m div 10;
     a3:=m mod 10; m:=m div 10;
     a2:=m mod 10; a1:=m div 10;
     {Проверка условий}
     If ((a1=a2) and (a1=a3) and
         (a1<>a4)) or {Первое условие}
        ((a1=a2) and (a1=a4) and
         (a1<>a3)) or {Второе условие}
        ((a1=a3) and (a1=a4) and
         (a1<>a2)) or {Третье условие}
        ((a2=a3) and (a2=a4) and
         (a2<>a1))   {Четвертое условие}
     Then Writeln(i:5);
    End;
    Readln;
End.
```

Рассмотрим выполнение программы для числа 3733. Трассировка примера:

n	m	a1	a2	a3	a4
3733	3733	-	-	-	3
3733	373	-	-	3	3
3733	37	3	7	3	3

Для данного числа выполняется третье условие, поэтому число 3733 будет напечатано.

4. Среди четырехзначных чисел выбрать те, у которых все четыре цифры различны.

5. Дано четырехзначное число n. Выбросить из записи числа n цифры 0 и 5, оставив прежним порядок остальных цифр. Например, из числа 1509 должно получиться 19.

6. Натуральное число из n цифр является числом Армстронга, если сумма его цифр, возведенных в n-ю степень, равна самому числу (например, $153=1^3+5^3+3^3$). Получить все числа Армстронга, состоящие из трех и четырех цифр.

7. Дана последовательность из 20 целых чисел. Определить количество чисел в наиболее длинной подпоследовательности из подряд идущих нулей.

1.2. ЦИКЛЫ С УСЛОВИЯМИ

1.2.1. Длинные целые числа. Циклы с предусловием

Длинные целые числа

Кроме обычных целых чисел, можно использовать так называемые длинные целые числа, которые могут принимать значения из большего диапазона. Переменные соответствующего типа описываются посредством идентификатора Longint и могут принимать значения в диапазоне от −2 147 483 648 до 2 147 483 647.

Цикл с предусловием

Цикл с предусловием используется тогда, когда число повторений оператора цикла заранее не известно, а задается некоторое условие продолжения цикла.

Оператор цикла с предусловием

While <условие> **Do** <тело цикла>;

Примечание. К уроку можно подготовить таблицу с конструкцией оператора цикла с предусловием While.

Выполнение оператора цикла с предусловием начинается с проверки условия, записанного после слова While. Если оно соблюдается, то выполняется <тело цикла>, затем вновь проверяется условие и т.д. Как только при очередной проверке окажется, что условие не соблюдается, <тело цикла> выполняться не будет.

Примечание

1. Если <тело цикла> состоит из нескольких операторов, то они объединяются операторными скобками.

2. В теле цикла обязательно должен быть оператор, влияющий на соблюдение условия, в противном случае произойдет зацикливание.

Пример

Подсчитать количество цифр заданного натурального числа n.

Решение

Подсчет количества цифр начнем с последней цифры числа. На очередном шаге цикла увеличим счетчик цифр на единицу, а число уменьшим в 10 раз (тем самым мы избавляемся от последней цифры числа). Далее с получившимся числом проделаем ту же последовательность действий и т.д., пока число не станет равным нулю.

```
Program Example_8;
Var  m, n: Longint;
     k: Integer;     {счетчик цифр}
Begin
  Writeln(' Введите натуральное число');
  {вводим натуральное число n>0}
  Readln(n);m:=n;
  k:=0;
  While m<>0 Do
  {пока (While) число m<>0  делать(Do)}
  Begin
   Inc(k);{ k:=k+1;}
   m:=m div 10;
   {"выбрасываем" из числа последнюю цифру}
  End;
  Writeln('В числе ',n,' — ',k,' цифр');
  {вывод количества цифр}
  Readln;
End.
```

Трассировка примера

Рассмотрим выполнение этой программы в пошаговом режиме для числа 65387:

n	m	k
65387	65387	0
65387	6538	1
65387	653	2
65387	65	3
65387	6	4
65387	0	5

В результате работы программы на экране появится предложение:

```
В числе 65387 — 5 цифр
```

Пример

Дана непустая последовательность натуральных чисел, за которой следует 0. Найти порядковый номер наименьшего элемента последовательности.

Решение

Обозначим через x и i очередной элемент последовательности и его номер; min и k — минимальный элемент последовательности и его номер. Считывание элементов последовательности производится до тех пор, пока не будет введен 0, то есть пока x<>0. Начальное значение минимума определяется значением первого элемента последовательности. Очередное вводимое число (очередной элемент последовательности) требуется сравнивать с текущим значением минимума, и если текущее значение min окажется больше очередного элемента последовательности, то min нужно изменить, а номер очередного элемента последовательности — запомнить. Учитывая вышесказанное, составим программу:

```
Program Example_9;
Var  x, i, min, k :  Integer;
Begin
  Writeln('Введите первый элемент
           последовательности');
  Read(x); k:=1;
  min:=x; i:=2;
  While x<>0 Do
  Begin
    If x<min Then
    Begin min:=x; k:=i-1 End;
    Writeln('Введите ',i,' элемент
             последовательности' );
    Read(x);
    Inc(i);
 End;
 Writeln('Номер минимального элемента — ', k);
End.
```

Решение задач

1. Дана последовательность операторов:

```
a:=1; b:=1;
While a+b<8 Do
Begin a:=a+1; b:=b+2 End;
s:=a+b;
```

Сколько раз будет повторен цикл и какими будут значения переменных a, b и s после завершения этой последовательности операторов?

2. Какими будут значения переменных a и b после выполнения последовательности операторов:

```
a:=1; b:=1;
While a<=3 Do a:=a+1; b:=b+1;
```

3. Определите значение переменной s после выполнения следующих операторов:

а)
```
s:=0; i:=0;
  While i<5 Do Inc(i); s:=s+100 div i;
```

б)
```
s:=0; i:=1;
  While i>1 Do
  Begin s:=s+100 div i; dec(i) End;
```

4. В последовательности операторов для вычисления факториала f числа n содержится пять ошибок. Найдите эти ошибки.

```
k:=1; f:=0;
While k<n Do f=f*k
k:=k+1,
```

5. Найдите и исправьте ошибки в следующем фрагменте программы, определяющей для заданного натурального числа n число, записанное цифрами числа n в обратном порядке.

```
p:=n;
While p>=0 Do
Begin
  a:=a+p mod 10;
  p:=p div 10
End;
```

6. Найти сумму цифр числа*.

7. Найти старшую цифру числа.

8. Приписать по 1 в начало и в конец записи числа n. Например, из числа n=3456 надо получить 134561.

9. Поменять местами первую и последнюю цифры числа.

10. Найти количество четных цифр натурального числа.

11. Найти самую большую цифру целого числа.

12. Найти сумму цифр целого числа, больших 5.

13. Сколько раз данная цифра встречается в целом числе?

14. Составить программу, проверяющую, является ли последовательность из 10 целых чисел, вводимых с клавиатуры, возрастающей.

15. Составить программу, проверяющую, является ли заданное натуральное число палиндромом, то есть таким, десятичная запись которого читается одинаково слева направо и справа налево.

---

* **При решении задач 6—13 требуется написать программы.**

1.2.2. Цикл с постусловием

Повторение

1. Запишите конструкцию цикла с предусловием. В каких случаях применяется цикл с предусловием?

2. Далее представлен фрагмент программы вычисления количества цифр в заданном натуральном числе. Найдите в нем ошибки и исправьте их.

```
a:=n; ck:=0;
While a>=0; Do
Begin
  ck:=ck+1;
  a:=a div 10
End;
```

3. Каким условиям должны удовлетворять значения переменной k, чтобы следующие циклы были бесконечны:

```
While c<0 Do c:=c+k;
While k<>0 Do k:=k+1;
While k<>0 Do k:=k-2.
```

Для программной реализации циклических алгоритмов с неизвестным заранее числом повторений имеется еще один оператор — оператор цикла с постусловием, который имеет следующий вид:

Оператор цикла с постусловием

```
Repeat {повторять}
  <оператор 1>;
  <оператор 2>;
  ...
  <оператор n>;
Until {до тех пор, пока не} <условие>;
```

Этот оператор отличается от цикла с предусловием тем, что проверка условия производится после очередного выполнения тела цикла. Это обеспечивает выполнение цикла хотя бы один раз.

Обратите внимание на то, что данный оператор цикла предполагает наличие нескольких операторов в теле цикла, поэтому служебные слова **Begin** и **End** не нужны.

Последовательность операторов, входящих в тело цикла, выполняется один раз, после чего проверяется соблюдение условия, записанного следом за служебным словом Until. Если условие соблюдается, цикл завершается. В противном случае тело цикла выполняется еще раз, после чего снова проверяется соблюдение условия.

Задание

Придумать примеры из жизни, иллюстрирующие данную конструкцию.

Пример

Составить программу планирования закупки товара в магазине на сумму, не превышающую заданной величины.

Решение

Обозначим через x и k цену и количество товара, через p — заданную предельную сумму, через s — стоимость покупки. Начальное значение общей стоимости покупки s равно нулю. Значение предельной суммы считывается с клавиатуры. Необходимо повторять запрос цены и количества выбранного товара, вычислять его стоимость, суммировать ее с общей стоимостью и выводить результат на экран до тех пор, пока стоимость не превысит предельной суммы p.

```
Program Example_10;
Var x, k, p, s: Integer;
Begin
  Writeln('Предельная сумма — ');
  Readln(p);
  s:=0;
  Repeat
    Writeln('Введите цену товара и его
             количество');
    Readln(x,k);
    s:=s+x*k;
   Writeln('Стоимость покупки равна ',s);
  Until s>p;
  Writeln('Суммарная стоимость покупки
           превысила предельную сумму');
End.
```

Решение задач

1. Определить значение переменной s после выполнения следующих операторов:

```
s:=0; i:=1;
Repeat s:=s+5 div i; i:=1-1;
Until i<=1;
```

2. Произведение первых n нечетных чисел равно p. Сколько сомножителей взято?

3. Числа Фибоначчи f_n определяются по формулам: $f_0=f_1=1$; $f_n=f_{n-1}+f_{n-2}$ при $n=2, 3, \ldots$

Составить программу для:

a) вычисления f_{40};

b) поиска первого числа Фибоначчи, большего m ($m>1$);

c) вычисления суммы всех чисел Фибоначчи, не превосходящих 1000.

4. Составить программу для определения того, является ли заданное натуральное число совершенным. *Совершенным называется число, равное сумме всех своих положительных делителей* (включая единицу, но исключая, разумеется, само число).

5. Показать, что любой оператор цикла с предусловием можно записать с помощью условного оператора и оператора цикла с постусловием.

6. Показать, что любой оператор цикла с постусловием можно записать с помощью условного оператора и оператора цикла с предусловием.

7. Дана непустая последовательность натуральных чисел, за которой следует 0. Вычислить сумму положительных элементов последовательности, порядковые номера которых нечетны.

1.2.3. Алгоритм Евклида

Повторение

1. Какие циклы с условиями вы знаете?

2. В чем сходство и в чем различие этих циклов?

Алгоритм Евклида — это алгоритм нахождения наибольшего общего делителя (НОД) двух целых неотрицательных чисел.

Пусть x и y — одновременно не равные нулю целые неотрицательные числа, и пусть $x \geq y$. Если $y=0$, то НОД$(x, y)=x$, а если $y\neq 0$, то для чисел x, y и r, где r — остаток от деления x на y, выполняется равенство НОД(x, y)=НОД(y, r).

Например, пусть $x=48$, а $y=18$.

НОД(48,18)=НОД(18,12)=НОД(12,6)= =НОД(6,0)=6.

Пример

Написать программу нахождения наибольшего общего делителя двух неотрицательных чисел.

Решение

Для решения данной задачи воспользуемся циклом с постусловием:

```
Program Example_11;
Var x, y: Integer;
Begin
  Writeln('Введите два числа');
  Readln(x,y);
  Repeat {выполнять}
  If x>y Then x:=x mod y Else y:=y mod x;
  Until (x=0) or (y=0);
  {до тех пор, пока одно из чисел
  не станет равно нулю}
  Writeln('НОД=', x+y));  {вывод НОД.
  Одно из чисел обязательно равно нулю}
  Readln;
End.
```

Пример

Даны натуральные числа x и y, не равные нулю одновременно. Найти d=НОД(x, y) и такие целые q и w, что $d=q*x+w*y$.

Решение

Введем переменные p, q, r, s, m и n, такие, что m=p*a+q*b, n=r*a+s*b. Первоначально m=a=x, n=b=y.

```
Program Example_12;
Var x,y: Integer;    {исходные данные}
    p,q,r,s,m,n: Integer;
     {введенные вспомогательные переменные}
    к: Integer;{для изменения значений
                p,q,r,s}
    d: Integer;{значение наибольшего
                общего делителя}
Begin
  Read(x,y);
  m:=x; n:=y; p:=1; q:=0; r:=0; s:=1;
  Repeat
    If m>n Then
    Begin
     k:=m div n; m:=m mod n;
     p:=p-k*r; q:=q-k*s
    End
    Else
    Begin
     k:=n div m; n:=n mod m; r:=r-k*p;
     s:=s-k*q
    End;
  Until (m=0) or (n=0);
  If m=0 Then Begin d:=n; q:=r; w:=s; End
  Else Begin d:=m; q:=p; w:=q; End;
  Writeln(d,'=',q,'*',x,'+',w,'*',y);
End.
```

Значения переменных p, q, r, s изменяются следующим образом:

- как только значение переменной m уменьшается на к*n, значение p уменьшается на к*r, а q уменьшается на к*s;
- аналогично, как только значение n уменьшается на к*m, значения переменных r и s уменьшаются соответственно на к*p и на к*q.

Решение задач

1. Найти НОД трех чисел.

Примечание. НОД(a,b,c)=НОД(НОД(a,b),c).

2. Два числа называются *взаимно простыми*, если их наибольший общий делитель равен 1. Проверить, являются ли два данных числа взаимно простыми.

3. Найти наименьшее общее кратное (НОК) чисел n и m, используя соотношение

$$\text{НОК}(n,m)=\frac{nm}{\text{НОД}(n,m)}.$$

4. Даны натуральные взаимно простые числа n, p. Найдите такое m, что, во-первых, $m<p$, во-вторых, произведение чисел m и n при делении на p дает остаток 1.

5. От прямоугольника 324×141 отрезают квадраты со сторонами 141, пока это возможно. Затем вновь отрезают квадраты со стороной, равной 324−2×141=42, и т.д. На какие квадраты и на сколько квадратов будет разрезан прямоугольник?

6. Написать программу для нахождения НОД, используя следующие соотношения:

НОД$(2a, 2b)$=2НОД(a, b);

НОД$(2a, b)$=НОД(a, b) при нечетном b.

В программе не должно использоваться деление с остатком. Можно лишь делить на 2 и проверять числа на четность.

7. Даны натуральные числа m и n. Найти такие натуральные взаимно простые p и q, что $\frac{p}{q}=\frac{m}{n}$.

1.2.4. Вложенные циклы

Повторение

1. Какие виды циклов вы знаете? В каких случаях применяется каждый из этих циклов?

2. Сколько раз выполнится тело цикла в следующих случаях:

a) `For i:=1 To 10 Do x:=x+i;`

b)
```
For k:=2 To 22 Do
   If k mod 2=0 Then s:=s+1;
```

```
c) For x:=-5 To 5 Do
    Begin
     Writeln('введите число');
     Readln(y);
     Writeln('x+y=',x+y);
    End;
```

3. Составить фрагмент программы возведения числа x в степень n.

Пример

Даны натуральные числа n и k. Составить программу вычисления выражения $1^k+2^k+...+n^k$.

Решение

Для вычисления указанной суммы целесообразно организовать цикл с параметром i, в котором, во-первых, находилось бы значение очередного члена ряда (y:=i^k) и, во-вторых, осуществлялось бы накопление искомой суммы путем прибавления полученного слагаемого к сумме всех предшествующих (s:=s+y).

```
Program Example_13;
Var n, k, y, i, s, m: Integer;
Begin
  Writeln('Введите n  и k ');
  Readln(n,k);
  s:=0;
  For i:=1 To n Do
  Begin
    y:=1;
    For m:=1 To k Do y:=y*i;
      {нахождение степени k числа i}
    s:=s+y;
  End;
  Writeln('Ответ: ',s);
End.
```

Для решения задачи потребовалось организовать два цикла, один из которых пришлось поместить внутрь другого. Такие конструкции называют *вложенными циклами*.

Пример

Модифицировать предыдущую программу так, чтобы в ней вычислялась сумма $1^1+2^2+...+n^n$.

Решение

Данная задача отличается от предыдущей тем, что показатель степени очередного слагаемого совпадает со значением ее основания, следовательно, верхняя граница внутреннего цикла (в котором вычисляется очередное слагаемое) определяется значением счетчика внешнего цикла.

```
Program Example_14;
Var n, y, i, s, m: Integer;
Begin
  Writeln('Введите значение n ');
  Readln(n);
  s:=0;
  For i:=1 To n Do
  Begin y:=1;
    For m:=1 To i Do y:=y*i;
    {нахождение степени k числа i}
    s:=s+y;
  End;
  Writeln('Ответ: ',s);
End.
```

Пример (старинная задача)

Сколько можно купить быков, коров и телят, если плата за быка — 10 рублей, за корову — 5 рублей, за теленка — полтинник (0,5 рубля) и на 100 рублей надо купить 100 голов скота?

Решение

Обозначим через b количество быков; k — количество коров; t — количество телят. После этого можно записать два уравнения: 10b+5k+0.5t=100 и b+k+t=100. Преобразуем их: 20b+10k+t=200 и b+k+t=100.

На 100 рублей можно купить:

- не более 10 быков, т.е. 0≤b≤10;
- не более 20 коров, т.е. 0≤k≤20;
- не более 200 телят, т.е. 0≤t≤200.

Таким образом, получаем:

```
Program Example_15;
Var b, k, t: Integer;
Begin
  For b:=0 To 10 Do
  For k:=0 To 20 Do
  For t:=0 To 200 Do
  If (20*b+10*k+t=200) and (b+k+t=100) Then
  Writeln('быков ',b,' коров ',k,' телят ',t);
End.
```

Сколько раз будет проверяться условие в данной программе?

Значение переменной b изменяется 11 раз (от 0 до 10), для каждого ее значения переменная k изменяется 21 раз, а для каждого значения переменной k переменная t изменяется 201 раз. Таким образом, условие будет проверяться 11×21×201 раз. Но если известно количество быков и коров, то количество телят можно вычислить по формуле t=100-(b+k) — и цикл по переменной t исключается.

```
Program Example_16;
Var b, k, t: Integer;
Begin
  For b:=0 To 10 Do
  For k:=0 To 20 Do
  Begin
    t:=100-(b+k);
    If (20*b+10*k+t=200) Then
   Writeln('быков ',b,' коров ',k,' телят ',t);
  End;
End.
```

При этом решении условие проверяется 11×21 раз. Можно ли еще уменьшить количество проверок?

Решение задач

1. Что будет напечатано после выполнения следующего фрагмента программы при n=6?

```
a:=1; b:=1;
For i:=0 To n Do
Begin
 For j:=1 To b Do Write('*');
 Writeln;
 c:=a+b; a:=b; b:=c;
End;
```

Решение какой задачи реализовано в этом фрагменте?

2. Что будет напечатано после выполнения следующего фрагмента программы при a=13305?

```
b:=0;
While a<>0 Do
Begin
  b:=b*10+a mod 10;
  a:=a div 10;
End;
Write(b);
```

Решение какой задачи реализовано в этом фрагменте?

3. Написать программу для нахождения всех прямоугольников, площадь которых равна заданному натуральному числу q и стороны выражены натуральными числами.

4. Составить программу для графического изображения делимости чисел от 1 до n (n задается). В каждой строке надо печатать очередное число и столько плюсов, сколько делителей у этого числа. Например, если задано число 4, то на экране должно быть напечатано:

```
1+
2++
3++
4+++
```

5. Составить программу получения всех совершенных чисел, меньших заданного числа n. Напомним, что число называется совершенным, если оно равно сумме всех своих положительных делителей, кроме самого себя. Например, 28 — совершенное число, т.к. 28=1+2+4+7+14.

Из истории. Грекам были известны первые четыре совершенных числа: 6, 28, 496, 8128. Эти числа высоко ценились. Даже в XII веке церковь утверждала, что для спасения души необходимо найти пятое совершенное число. Это число было найдено только в XV веке. До сих пор совершенные числа полностью не исследованы — неизвестно, имеется ли конечное число совершенных чисел или их число бесконечно, кроме того, неизвестно ни одного нечетного совершенного числа, но и не доказано, что таких чисел нет.

6. Дано натуральное число n. Можно ли его представить в виде суммы квадратов трех натуральных чисел? Если можно, то:

a) указать тройку x, y, z таких натуральных чисел, что $x^2+y^2+z^2=n$;

b) указать все тройки x, y, z таких натуральных чисел, что $x^2+y^2+z^2=n$.

7. Найти натуральное число от 1 до 10 000 с максимальной суммой делителей.

8. Даны натуральные числа a, b ($a<b$). Получить все простые числа p, удовлетворяющие неравенству $a \le p \le b$.

9. Даны натуральные числа n, m. Получить все натуральные числа, меньшие n, квадрат суммы цифр которых равен m.

10. Даны натуральные числа n и m. Найти все натуральные числа пары дружественных чисел, лежащих в диапазоне от n до m. Два числа называются дружественными, если каждое из них равно сумме всех делителей другого (само число в качестве делителя не рассматривается).

11. Переставить цифры данного натурального числа таким образом, чтобы образовалось наименьшее число, записанное этими цифрами.

12. Составить программу, печатающую k-ю цифру последовательности:

a) 12345678910..., в которой выписаны подряд все натуральные числа;

b) 14916253649..., в которой выписаны подряд квадраты всех натуральных чисел;

c) 1123581321..., в которой выписаны подряд все числа Фибоначчи.

13. Составить программу возведения заданного числа в третью степень, используя следующую закономерность:

$1^3=1$
$2^3=3+5$
$3^3=7+9+11$
$4^3=13+15+17+19$
$5^3=21+23+25+27+29$

14. Составить программу для нахождения всех натуральных решений уравнения $n^2+m^2=k^2$, где n, m и k лежат в интервале $[1, 10]$.

Примечание. Решения, которые получаются перестановкой n и m, считать совпадающими.

1.2.5. Решение задач с использованием циклов с условием

Повторение

1. Написать фрагмент программы для решения указанной ниже задачи и обосновать, почему был выбран тот или иной вариант оператора цикла:

a) вычислить факториал числа p;

b) факториал некоторого числа равен p, найти само число;

c) определить, является ли заданное число степенью числа 3;

d) вычислить $y=1!+2!+3!+...+n!$

e) вычислить x^n, где n — целое положительное число;

f) вычислить значения $x^1, x^2, x^3, ..., x^n$.

Пример

Написать программу, которая находит и выводит на печать все четырехзначные числа $\overline{abcd}$, где a, b, c, d — различные цифры, для которых выполняется соотношение: $\overline{ab} - \overline{cd} = a + b + c + d$.

Решение

Задачу можно решать несколькими способами. Можно просто перебрать все четырехзначные числа и для каждого проверять соблюдение условий. Попробуем сократить перебор, для этого преобразуем второе условие:

$\overline{ab} - \overline{cd} = a + b + c + d$;

$10a+b-(10c+d)=a+b+c+d$;

$9(a-c)=2(c+d)$;

$$\frac{a-c}{c+d}=\frac{2}{9}.$$

Учитывая первое условие (a, b, c, d — различные цифры), получаем, что $a=c+2$, $d=9-c$ и $0\le c\le 7$.

```
Program Example_17;
Var a,b,c,d:Integer;
Begin
  For c:=0 To 7 Do
  Begin
    a:=c+2; d:=9-c;
    For b:=0 To 9 Do
    If (b<>c) and (b<>a) and (b<>d)
    Then Write(a,b,c,d,' ');
    Writeln
  End;
End.
```

Таким образом, мы решили задачу, значительно сократив перебор.

Пример

Если мы сложим все цифры какого-либо числа, затем все цифры найденной суммы и будем повторять этот процесс, мы наконец получим однозначное число (цифру), называемое *цифровым корнем* данного числа. Например, цифровой корень числа 34 697 равен 2 (3+4+6+9+7=29; 2+9=11; 1+1=2). Составим программу для нахождения цифрового корня натурального числа.

Решение

Вопросы для обсуждения:

1. Сколько переменных потребуется для решения задачи, какого типа будут эти переменные?

2. Всегда ли необходимо вычислять сумму цифр числа (а если введенное число является однозначным)?

3. Вычислим сумму цифр числа: для этого будем выделять цифры числа и увеличивать текущую сумму. Какую конструкцию необходимо использовать для этого?

4. В результате выполнения цикла мы получили число. Является ли оно однозначным?

Программа, вычисляющая цифровой корень данного числа, может выглядеть следующим образом:

```
Program Example_18;
Var n,k,s:Longint;
Begin
  Writeln('Введите число');
  Readln(n);
  s:=n;
  While s>9 Do
  Begin
    k:=s; s:=0;
    Repeat
      s:=s+k mod 10;
      k:=k div 10;
    Until k=0;
  End;
  Writeln('Цифровой корень числа ',n,' равен ',s);
End.
```

Решение задач

1. Найти все трехзначные числа, удовлетворяющие следующим условиям:

a) любые две цифры числа различны;

b) число равно среднему арифметическому всех трехзначных чисел (включая данное), состоящих из тех же цифр.

2. Стороны прямоугольника заданы натуральными числами M и N. Составить программу, которая находит, на сколько квадратов, стороны которых выражены натуральными числами, можно разрезать данный прямоугольник, если от него каждый раз отрезается квадрат максимальной площади.

3. Дано натуральное число $n\ge 2$. Составить программу разложения этого числа на простые множители. Реализовать два варианта:

a) каждый простой множитель p должен быть выведен k раз, где k — натуральное число, такое, что n делится на p^k и не делится на p^{k+1};

b) каждый простой множитель должен быть выведен ровно один раз.

4. Даны натуральные числа N и p. Получить все натуральные числа, меньшие N и взаимно простые с p.

5. Даны целые числа p и q. Получить все делители числа q, взаимно простые с p.

6. Найти наименьшее натуральное число N, представимое двумя различными способами в виде суммы кубов двух натуральных чисел.

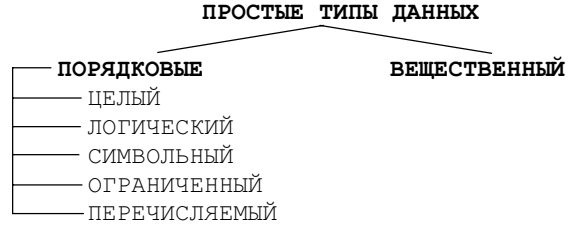
7. Даны натуральные числа m, n_1, n_2, ..., n_m ($m\ge 2$). Вычислить НОД(n_1, n_2, ...,n_m), используя соотношение НОД(n_1, n_2, ..., n_m)=НОД(НОД(n_1, n_2, ...,n_{m-1}), n_m) и алгоритм Евклида.

8. Найти все простые несократимые дроби, заключенные между 0 и 1, знаменатели которых не превышают 7 (дробь задается двумя натуральными числами — числителем и знаменателем).

1.3. ПРОСТЫЕ ТИПЫ ДАННЫХ

1.3.1. Символьный тип данных

Простые типы данных делятся на порядковые и вещественный типы данных.



В математике *порядковым числом* называется номер элемента при перечислении. Под *порядковым типом* понимают тип данных, областью значений которых является упорядоченное счетное множество. Каждому элементу такого множества соответствует некоторое порядковое число, как раз и являющееся его номером при перечислении.

В любом порядковом типе для каждого значения, кроме первого, существует предшествующее значение, и для каждого значения, кроме последнего, существует последующее значение. В языке Паскаль имеются стандартные функции, которые позволяют определять предшествующее и последующее значения для заданного значения:

функция `Pred(x)` определяет значение, предшествующее `x`;

функция `Succ(x)` определяет значение, следующее за `x`;

функция `Ord(x)` возвращает порядковый номер величины `x`.

К порядковым типам относятся целый и логический, а также символьный типы данных.

Символьный тип данных

Данные символьного типа описываются с помощью идентификатора `Char`.

Значением переменной символьного типа может быть любой символ — буквы, цифры, знаки препинания и специальные символы. Каждому символу алфавита соответствует индивидуальный числовой код от 0 до 255.

Примечание. Наиболее распространенной международной системой кодирования символов является система ASCII. Символы с кодами от 0 до 127 составляют так называемую основную таблицу кодов ASCII. Эта часть идентична на всех IBM-совместимых компьютерах. Символы с кодами от 128 до 255 составляют так называемую национальную кодовую таблицу. Именно в ней располагаются, например, русские буквы.

К символьным данным применимы операции сравнения. Операция сравнения осуществляется следующим образом: из двух символов меньше тот, который встречается в таблице ASCII раньше.

Обычно значения для переменных типа `Char` задаются в апострофах: `ch:='*'; a:='3'; letter:='G'`.

Кроме того, имеется возможность задавать значения непосредственно указанием ASCII-кода: `kd:=#65 {символ 'A'}; s:=#13 {код клавиши` Enter `}`.

Первые 32 символа ASCII являются управляющими. Для указания этих символов используются записи `#<ASCII-код>` или `^<символ>`. Например, `^[` — символ, соответствующий клавише Esc; `^G` — звуковой сигнал.

Так как символьный тип является порядковым, то для него справедливо все, что было сказано о порядковых типах.

Задание

1. Напишите программу, которая запрашивает символ и выводит на экран его код.

2. Функция `Chr(x)` возвращает символ по его коду. Напишите программу для вывода на экран всех символов таблицы ASCII.

Пример

Написать программу вывода последовательности символов `AABABC...AB...YZ`.

Решение

Последовательность символов строится по следующему правилу: последовательно выводятся начальные отрезки латинского алфавита, состоящие из 1 символа, из 2 символов и так далее, до тех пор, пока не будет выведен весь алфавит. Количество таких отрезков равно количеству букв в алфавите. Так как символьный тип данных является порядковым типом, то можно использовать цикл с параметром символьного типа:

```
Program Example_19;
Var i,j:Char;
Begin
  For i:='a' To 'z' Do
  {количество начальных отрезков алфавита}
    For j:='a' To i Do
    {количество символов в данном начальном отрезке}
    Write(j);
   Readln;
End.
```

Задание

Модифицировать программу для вывода последовательности символов в виде пирамиды:

a)

```
          a
        a b
      a b c
        ...
  a b c... x y z
```

b)

```
          a
        a b
      a b c
        ...
  a b c d... x y z
```

Пример

Написать программу, которая подсчитывает количество цифр, входящих в исходный текст. Текст — это последовательность символов, ввод которой заканчивается нажатием клавиши Enter.

Решение

Так как признаком конца ввода последовательности символов служит нажатие клавиши Enter (ей соответствует символ с кодом 13), то будем вводить символы до тех пор, пока значение очередного символа не совпадет со значением #13. Анализируя каждый символ, будем увеличивать счетчик, если символ является цифрой:

```
Program Example_20;
Var ch:Char;
    k:Integer;
Begin
  Read(ch);
  k:=0;
  While ch<>#13 Do
  {пока не нажата клавиша <Enter>}
  Begin
   If (ch>='0') and (ch<='9') Then Inc(k);
   Read(ch);
  End;
  Writeln(^G,'Количество цифр равно ',k);
End.
```

Задание

Модифицировать программу так, чтобы с ее помощью можно было:

a) определить, является ли текст правильной записью целого числа;

b) вычислить сумму цифр введенного числа.

Переменные символьного типа удобны для организации простейшего диалога с пользователем во время выполнения программы.

Приведем пример такого диалога:

```
Repeat
   ...
  Writeln('Хотите продолжить работу(y/n)?');
   Readln(ch);
Until Upcase(ch)='N';
```

Функция `Upcase(ch)` преобразует букву `ch` в прописную (эта функция работает только с буквами английского алфавита, все другие символы не изменяются).

Решение задач

1. Написать программу вывода последовательности символов:

a) `ZYYXXX...AA...AA`;

b) `ABC...ZZBC...ZZZC...ZZ...ZZ`.

2. Составить программу, которая печатает `true`, если во введенном тексте буква `A` встречается чаще, чем `B`, и печатает `false` в противном случае.

3. Проверить, правильно ли в заданном тексте расставлены круглые скобки (т.е. находится ли справа от каждой открывающей скобки соответствующая ей закрывающая скобка, а слева от каждой закрывающей — соответствующая ей открывающая).

4. Дана последовательность символов, имеющая следующий вид: $d_1 \pm d_2 \pm \ldots \pm d_n$ (все d_i — цифры, $n>1$). Вычислить значение выражения.

5. Составить программу, запрашивающую координаты ферзя на шахматном поле и показывающую поля доски, находящиеся под боем.

6. Составить программу, которая выводит на экран следующие изображения:

a)

```
**************
*            *
*            *
**************
```

b)

```
    *
   * *
  *   *
 *     *
*********
```

1.3.2. Вещественный тип данных

Повторение

1. Что будет выведено на экран в результате исполнения одного из следующих фрагментов программы?

a) `a:='Э';b:='B'; c:='M'; write(a,b,c,#13);`

b) `a:='Э';b:='B'; c:='M'; writeln(a,b,c).`

2. Напишите фрагмент программы для вывода на экран следующего изображения:

```
1
 2
  3
   4
    5
     6
      7
```

3. Какую конструкцию цикла целесообразно использовать при решении следующих задач?

a) Вычислить *n!!* (*n!!*=1 × 3 × 5 × ... × *n* для нечетных *n* и *n!!*=2 × 4 × ... × *n* для четных *n*).

b) Для всех чисел от 1 до *n* вывести на экран значения *n!!*.

c) Для заданного числа найти *m*, для которого *m!!*=*n*.

Вещественный тип данных

Данные вещественного типа описываются с помощью идентификатора `Real`. Они могут принимать значения в диапазоне $2{,}9 \times 10^{-39}$... $1{,}7 \times 10^{38}$.

Константа действительного типа может быть представлена в двух видах: числом с фиксированной и плавающей точкой.

Число с фиксированной точкой изображается десятичным числом с дробной частью (дробная часть может быть нулевой). Дробная часть отделяется от целой с помощью точки, например: `127.3, 25.0, -16.003, 200.59, 0.54`.

Число с плавающей точкой имеет вид *mEp*, где *m* — мантисса, а *p* — порядок числа. В качестве *m* могут быть целые числа и действительные числа с фиксированной точкой, в качестве *p* — только целые числа. Как мантисса, так и порядок могут содержать знаки "+" и "−". Приведем примеры.

Математическая запись	*Запись с плавающей точкой*
0,000009	`9E-6`
$0{,}62 \cdot 10^4$	`0.62E+4`
$-10{,}8 \cdot 10^{12}$	`-10.8E12`
$20 \cdot 10^{-3}$	`20E-3`

В языке Паскаль имеется много стандартных функций для работы с вещественными числами. Перечислим наиболее часто используемые:

Abs(x)	абсолютное значение x
Sqr(x)	квадрат x
Sqrt(x)	квадратный корень из x
Sin(x)	синус x
Cos(x)	косинус x
Arctan(x)	арктангенс x
Exp(x)	e^x
Ln(x)	натуральный логарифм x
Trunc(x)	целая часть x
Round(x)	ближайшее к x целое число

Примечание. В тригонометрических функциях синус и косинус аргумент задается только в радианах.

Выводить данные вещественного типа можно по формату и без него. Если при выводе данных вещественного типа не указан формат, то число выводится с плавающей точкой — мантисса и порядок. На число отводится 17 позиций, при этом в целой части мантиссы присутствует только одна значащая цифра. Изменить стандартную форму вывода можно, используя формат `Write(x:m:n)`, где x — выводимое данное вещественного типа; m — общее количество позиций для вывода числа (включая знак числа, целую часть, точку и дробную часть); n — количество позиций для вывода дробной части. В качестве m и n могут быть целые константы, переменные, выражения. Так, использование формата `Write(r:8:4)` для вывода значения r, равного `-35.245367`, приводит к выводу значения `-35.2454`.

Задание

Исследовать, что произойдет, если:

a) в формате указано большее количество позиций, чем необходимо;

b) число не помещается в заданный формат (т.е. если число действительных разрядов в переменной окажется больше числа разрядов, выделенных для вывода переменной на экран);

c) дробная часть числа не укладывается в заданный формат.

Пример

Напечатать таблицу значений функции $\sin x$ на отрезке [0,1] с шагом 0,1 (каждое вещественное число вывести в четырех позициях на отдельной строке).

Решение

Для перебора всех значений из отрезка с некоторым шагом проще всего использовать цикл с параметром. Но нам требуется перебирать вещественные числа, а параметр цикла не может быть вещественным. Поэтому придется использовать цикл `While`.

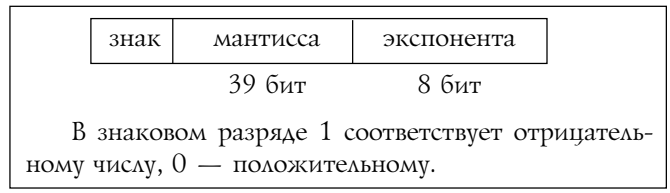
Сравните два решения этой задачи, представленные ниже.

```
Program Example_21;
Var  i: Real;
Begin
  i:=0;
  While i<=1 Do
  Begin
    Writeln(i:2:1,' ',sin(i):4:3);
    i:=i+0.1;
  End;
  Readln;
End.
```

```
Program Example_22;
Var i: Integer;
Begin i:=0;
    While i<=10 Do
    Begin
     Writeln(i,' ',sin(i/10):4:3);
     Inc(i);
    End;
   Readln;
End.
```

На первый взгляд эти программы должны работать одинаково, но, запустив их, мы обнаружим, что первая программа выдает значения функции для всех значений x от 0 до 0.9, а вторая программа — для всех значений x от 0 до 1.

Почему это происходит? Как вы знаете, вся информация представляется в памяти ЭВМ в виде 0 и 1. Для хранения переменной типа `Real` в памяти ЭВМ отводится 48 бит (6 байт), которые распределяются следующим образом:



В знаковом разряде 1 соответствует отрицательному числу, 0 — положительному.

Рис. 4

Переведем вещественное число 0,1 в двоичную систему, для этого будем умножать число на 2:

0,1 × 2=0,2
0,2 × 2=0,4
0,4 × 2=0,8
0,8 × 2=1,6
0,6 × 2=1,2
0,2 × 2=0,4,

— далее вычисления повторяются.

Выписывая подчеркнутые цифры, получим двоичную дробь:

$0{,}1_{10}=0{,}0001100110011..._2$.

Можно заметить, что полученная дробь является периодической с периодом 0011. Согласно *рис.* 4 мантисса 00000110011... обрывается после 40 разрядов. Таким образом, получаем приближенное представление вещественного числа.

Вернемся к нашему примеру. Сейчас можем объяснить, почему первая программа работает не так, как нам хотелось бы, — это происходит из-за приближенного представления вещественного числа — последнее значение получается чуть больше 1. Поэтому при решении задач следует избегать использования циклов, в условиях которых используются вещественные переменные.

Пример

Дано x, принадлежащее интервалу от -1 до 1. Составить программу вычисления частичной суммы ряда $x-\frac{x^2}{2}+\frac{x^3}{3}-\ldots$ с заданной точностью e. Нужная точность считается полученной, если очередное слагаемое оказалось по модулю меньше, чем e (это и все последующие слагаемые учитывать не надо).

Вопросы для обсуждения

1. Как можно запрограммировать смену знака слагаемого?

2. Какой конструкцией необходимо воспользоваться для получения нужной точности приближения?

3. Сколько переменных необходимо для решения задачи, каковы их начальные значения?

```
Program Example_23;
Var x,st,sl,y,e:Real;
    n,z:Integer;
Begin
 Write('Введите x, принадлежащее (-1,1) ');
 Readln(x);
 Write('Введите погрешность вычисления ');
 Readln(e);
 y:=0; n:=1; z:=1; st:=x; sl:=x;
 Repeat
   Inc(y,z*sl); Inc(n); z:=-z;
   st:=st*x; sl:=st/n;
 Until sl<e;
 Writeln(y);
 Readln;
End.
```

Решение задач

1. Даны действительные положительные числа a, b, c, x, y. Выяснить, пройдет ли кирпич с ребрами a, b, c в прямоугольное отверстие со сторонами x, y. Просовывать кирпич разрешается только так, чтобы каждое из его ребер было перпендикулярно или параллельно каждой из сторон отверстия.

2. Написать фрагмент программы и обосновать выбор того или иного варианта оператора цикла:

a) вычислить c — наибольший общий делитель натуральных чисел a и b;

b) найти u — первый отрицательный член последовательности $\cos(\operatorname{ctg} n)$, где $n=1, 2, 3, \ldots$;

c) вычислить $p=(1-\frac{1}{2^2})(1-\frac{1}{3^2})\ldots(1-\frac{1}{n^2}),\ n>2$;

d) вычислить $y=\cos(1+\cos(2+\ldots+\cos(39+\cos 40)\ldots))$.

3. Задаются коэффициенты квадратного уравнения $ax^2+bx+c=0$. Составить программу для нахождения его корней.

4. Вычислить значение выражений:

a) $\sin x+\sin\sin x+\ldots+\underbrace{\sin\sin\ldots\sin x}_{n\text{ раз}}$;

b) $\sin x+\sin x^2+\ldots+\sin x^n$;

c) $\sin x+\sin^2 x+\ldots+\sin^n x$;

d) $\frac{\cos 1}{\sin 1}\cdot\frac{\cos 1+\cos 2}{\sin 1+\sin 2}\cdot\ldots\cdot\frac{\cos 1+\ldots+\cos n}{\sin 1+\cdots+\sin n}$.

5. Представить обыкновенную дробь $\frac{m}{n}$ $(m<n)$ в виде цепной дроби

$$\frac{m}{n}=\frac{1}{a_1+\frac{1}{a_2+\ldots\frac{1}{a_k}}}.$$

Например, $\frac{5}{7}=\frac{1}{1+\frac{1}{2+\frac{1}{2}}}$.

6. Вычислить $\cfrac{1}{1+\cfrac{1}{3+\cfrac{1}{5+\cfrac{1}{\ldots+\cfrac{1}{101+\cfrac{1}{103}}}}}}$.

7. Рассмотрим бесконечную последовательность $y_1, y_2, y_3, \ldots$, образованную по следующему закону:

$$y_1=\frac{x+m-1}{2},\ y_i=\frac{1}{m}\left((m-1)y_{i-1}+\frac{x}{y_{i-1}^{m-1}}\right),\ (i=2, 3,\ldots),$$

где x — данное действительное число, m — натуральное число. Эта последовательность стремится к пределу, равному $\sqrt[m]{x}$ $(x\ge 0)$. Составить программу для вычисления значения $\sqrt[m]{x}$ с заданной точностью e $(e=|x-y_i^m|)$.

8. Вычислить $1-\frac{1}{2}+\frac{1}{3}-\ldots+\frac{1}{9999}-\frac{1}{10000}$ четырьмя способами:

a) последовательно слева направо;

b) последовательно слева направо вычисляются $1+\frac{1}{3}+\ldots+\frac{1}{9999}$ и $\frac{1}{2}+\frac{1}{4}+\ldots+\frac{1}{10000}$, затем второе значение вычитается из первого;

c) последовательно справа налево;

d) последовательно справа налево вычисляются суммы, указанные в задании **b**, затем вторая вычитается из первой.

Почему при вычислениях каждым из этих способов получаются разные результаты?

9. Вычислить с заданной точностью e

a) $\frac{1}{1\times 2}+\frac{1}{2\times 3}+\frac{1}{3\times 4}+\ldots$;

b) $\frac{1}{1\cdot 3}+\frac{1}{2\cdot 4}+\frac{1}{3\cdot 5}+\ldots;$

c) $\frac{1}{1\cdot 2\cdot 3}+\frac{1}{2\cdot 3\cdot 4}+\frac{1}{3\cdot 4\cdot 5}+\ldots$

10. Дано действительное число x. Вычислить с точностью e частичную сумму ряда.

a) $\sin x=\frac{x}{1!}-\frac{x^3}{3!}+\frac{x^5}{5!}-\frac{x^7}{7!}-\ldots(-1)^n\frac{x^{2n+1}}{(2n+1)!}+\ldots$

b) $\cos x=1-\frac{x^2}{2!}+\frac{x^4}{4!}-\frac{x^6}{6!}+\ldots+(-1)^n\frac{x^{2n}}{(2n)!}+\ldots$

Нужное приближение считается полученным, если очередное слагаемое оказалось по модулю меньше e.

11. Составить программу перевода действительного числа a $(0<a<1)$ в двоичную систему счисления.

12. Дано 10 вещественных чисел. Вычислить разность между максимальным и минимальным из них.

13. Дано 20 вещественных чисел. Определить, сколько из них больше своих "соседей", то есть предыдущего и последующего чисел.

1.3.3. Ограниченный и перечисляемый типы данных. Оператор варианта

Повторение

1. Определите значение переменной s после выполнения следующих операторов:

a)
```
s:=0; i:=0;
While i<5 Do Inc(i); s:=s+1/i;
```

b)
```
s:=0; i:=1;
While i>1 Do
Begin s:=s+1/i; dec(i) End;
```

c)
```
s:=0; i:=1;
Repeat s:=s+1/i; dec(i) Until i<=1;
```

d)
```
s:=1; i:=1;
For i:=2 To n Do s:=s+1/i.
```

2. Какие порядковые типы данных вы знаете? Какими общими свойствами они обладают?

Ограниченный тип данных

Ограниченный тип данных представляет собой интервал значений порядкового типа, называемого базовым типом. Описание типа задает наименьшее и наибольшее значения, входящие в этот интервал. Например,

```
Var a:1..25; ch: 'a'..'z';
```

Здесь переменные a и ch могут принимать значения только из указанного интервала; базовым типом для переменной a является целый тип, а для переменной ch — символьный.

Переменная ограниченного типа сохраняет все свойства переменных базового типа.

Для чего применяется ограниченный тип данных? Использование ограниченного типа делает программу более наглядной и понятной. Например, если в программе переменная b может принимать только значения 3, 4, 5, 6, 7, 8, то лучше описать ее следующим образом: **Var** b:3..8; а не просто **Var** b: Integer; тогда в случае выхода значения b за диапазон 3..8 при использовании первого описания будет выдано диагностическое сообщение, которое поможет найти ошибку.

Пример

Напишите программу, в результате выполнения которой переменной t присваивается значение true, если первая дата предшествует (в рамках одного года) второй дате, и значение false в противном случае.

Решение

Так как в условии задачи оговаривается, что обе даты должны находиться в рамках одного года, то дату достаточно задать указанием дня и месяца. Количество дней любого месяца года не может быть более 31, количество месяцев в году равно 12. Значение переменной t равно true, если номер первого месяца меньше второго, либо значение первого дня меньше второго при условии, что номера месяцев совпали.

```
Program Example_24;
Var  d1,d2:1..31;
     m1,m2:1..12;
     t:Boolean;
Begin
  Write('Введите первую дату (день, месяц) ');
  Readln(d1,m1);
  Write('Введите вторую дату (день, месяц) ');
  Readln(d2,m2);
  t:=(m1<m2) or ((m1=m2) and (d1<d2));
  Writeln(t);
End.
```

Задание

Модифицировать программу так, чтобы в ней осуществлялась проверка корректности введенных дат.

Оператор варианта (выбора)

Оператор варианта

```
Case <порядковая переменная> Of
<константа 1>: <оператор 1>;
< константа 2>: <оператор 2>;
...
< константа n>: <оператор n>;
[Else <оператор>;]
End;
```

Выполнение оператора выбора начинается с вычисления выражения, которое должно принимать значение порядкового типа. В случае если результат вычисления равен одной из перечисленных констант, то выполняется соответствующий оператор. Затем управление передается за пределы оператора выбора. Если значение выражения не совпадает ни с одной константой, то выполняется оператор, стоящий после Else, если он есть, либо управление передается на оператор, следующий за End.

Эта конструкция предназначена для замены конструкций из вложенных условных операторов и применяется для обработки ситуаций с несколькими исходами.

Примечание

1. Тип константы должен совпадать с типом выражения.
2. Ветвь `Else` заключена в квадратные скобки, что говорит о том, что эта часть оператора выбора необязательна.
3. В конструкции выбора, в отличие от условного оператора, перед `Else` ставится точка с запятой.
4. В качестве операторов могут использоваться и составные операторы.
5. Можно задавать не только одну константу, но и список и диапазон констант. Соответствующие примеры приведены далее.

Пример

Составьте программу, в которой определяется, какой буквой — гласной или согласной — является введенный символ английского алфавита.

Решение

Разделим все символы на три группы:

- гласные буквы английского алфавита;
- согласные буквы английского алфавита;
- символы, не являющиеся буквами английского алфавита.

```
Program Example_25;
Var ch: Char;
Begin
  Write('Введите символ ');
  Readln(ch);
  Case Upcase(ch) Of
  'A','E','I','O','U': Writeln('Это
   гласная буква английского алфавита ');
  'A'..'Z': Writeln('Это согласная буква
   английского алфавита');
  Else Writeln('Этот символ не является
         буквой английского алфавита ');
End.
```

Обратите внимание на то, что в первом случае константы перечисляются через запятую, а во втором используется интервал значений.

Задание

Напишите эту программу, используя условный оператор.

Перечисляемый тип данных

Этот тип данных получил название перечисляемого, потому что он задается в виде перечисления некоторых значений. Эти значения образуют упорядоченное множество и являются константами. Для объявления переменной список возможных значений, разделенных запятой, указывается в круглых скобках. Например,

```
Var month:(january, february, march,
april, may, june, july, august,
september, october, november, december);
```

Порядок элементов перечисляемого типа определяется порядком их следования в описании. Левый имеет минимальное значение (значение функции `Ord` для него равно 0), а правый — максимальное.

Задание

Вычислить значения:

a) `Ord(august);`

b) `Ord(succ(september));`

c) `Pred(Pred(december)).`

К переменным перечисляемого типа можно применять операции сравнения. Так, например, `february <november.`

Использование перечисляемого типа данных повышает читабельность программы. Однако в языке Паскаль нельзя вводить и выводить элементы перечисляемого типа. Этот недостаток легко преодолевается, так как всегда можно ввести число, являющееся порядковым номером элемента перечисляемого типа, и вывести на экран значение переменной перечисляемого типа, используя, например, оператор выбора `Case`.

Задание

Написать фрагмент программы, формирующий значения переменной `month`, используя перечисляемый тип данных, внести соответствующие изменения в решение примера 1.

Решение задач

1. Имеются описания:

```
Var x,y: (winter, spring, summer,
          autumn);
  t: (cold, warm);
```

a) Допустимы ли присваивания:

```
x:=spring;t:=warm;t:=hot;y:=x;y:=t;
```

b) Вычислить значения выражений:

```
spring<summer;
autumn<winter;
Succ(spring);
Pred(autumn);
Ord(spring);
winter<=summer;
spring<>warm;
Pred(spring);
Pred(cold);
Pred(autumn)+Ord(cold);
```

c) Допустим ли оператор цикла с заголовком `For x:=spring To autumn Do?`

2. Напишите программу, которая по заданной дате определяет время года. Программа должна проверять корректность введенной даты.

3. Даны описания следующих переменных:

```
Var  m, m1: (january, february, march,
april, may, june, july, august,
september, october, november, december);
k: 1..maxint; n: 1..12;
```

Присвоить переменной m1:

a) название месяца, следующего за месяцем m;

b) название k-го месяца после месяца n.

4. В старояпонском календаре был принят 60-летний цикл, состоящий из пяти 12-летних подциклов. Подциклы обозначались названием цвета: зеленый, красный, желтый, белый и черный. Внутри каждого подцикла годы носили названия животных: крысы, коровы, тигра, зайца, дракона, змеи, лошади, овцы, обезьяны, курицы, собаки и свиньи. Напишите программу, которая по номеру года определяет его название по старояпонскому календарю (1984-й — год Зеленой Крысы).

5. Дано неотрицательное число k, не превышающее десяти тысяч. Напечатать фразу "k ворон", подставив вместо k соответствующее числительное. (Пример: при k=23 должно быть напечатано "двадцать три вороны"; при k=3651 — "три тысячи шестьсот пятьдесят одна ворона".)

6. Имеются описания:

```
Var d:'0'..'9'; k:0..9; n:Integer;
```

a) Какие значения может принимать переменная d? Каков ее базовый тип? Допустимы ли присваивания: d:='7'; d:='a'; d:=7?

b) Какие значения может принимать переменная k? Каков ее базовый тип? Допустимы ли присваивания: k:=5; k:=10; k:=-0; k:='5'?

c) Есть ли ошибки в операторе:

```
If k+n>7*k Then k:=abs(n) mod 10
           Else d:=chr(k+Ord('0')?
```

7. Имеются описания переменных:

```
Var k1, k2: (north, east, south,
             west);
pr:(forward, right, back, left);
```

Корабль сначала шел по курсу k1, а затем его курс был изменен согласно приказу pr. Определить k2 — новый курс корабля.

1.3.4. Описание переменных, констант и типов

Раздел описания констант

Константа — это величина, которая не изменяет своего значения в процессе выполнения программы. С константами мы уже встречались, так как в общем случае константой является любое целое или вещественное число, символ, идентификаторы false и true, а также идентификаторы, обозначающие значения переменных перечисляемого типа. Но константа может быть обозначена и именем. В этом случае она должна быть объявлена в разделе описания констант. Раздел описания констант начинается словом **Const** (от англ. *constancy* — постоянство).

Например,

```
Const N=25; K=38; D=(N+K) Div 2;
   Letter='f'; M=5E15;
```

Здесь N, K, D — целочисленные константы, Letter — константа символьного типа, а M — константа вещественного типа. Следует отметить, что константа D принимает свое значение после вычисления выражения. В разделе описания констант можно использовать лишь некоторые стандартные функции, такие, как Abs, Chr, Pred, Succ, Odd, Ord.

Именование констант делает программу более удобной для понимания и внесения исправлений. При изменении констант достаточно будет изменить соответствующие значения в разделе описания констант.

Раздел описания переменных

Ответьте на вопросы:

1. В каком разделе описываются переменные?

2. Чем характеризуется переменная?

Наряду с переменными и константами имеются и так называемые типизированные константы. В описании типизированной константы присутствуют описание типа и одно из допустимых значений, например,

```
Const N:Integer=15; ch:Char=#87;
```

Типизированные константы являются, собственно говоря, переменными. В частности, они могут изменять свое значение в процессе выполнения программы. От обычных переменных они отличаются лишь тем, что инициализируются при запуске программы.

Раздел описания типов

В языке Паскаль все данные, используемые программой, должны принадлежать к какому-либо заранее известному типу данных.

Тип данных определяет:

- формат представления данных в памяти ЭВМ;
- множество допустимых значений;
- множество допустимых операций.

Примечание. Следует отметить, что все типы данных изучались по этой схеме.

Все простые типы языка Паскаль можно разделить на стандартные и пользовательские. К стандартным типам относятся типы: Integer, Real, Char, Boolean, а также некоторые другие, описание которых приведено в табл. 2 и 3.

Пользовательские типы объявляются в разделе описания типов, который открывается словом **Type**.

Пример

```
Type
week=(sunday,monday,tuesday,wednesday,thursday,
     friday,saturday);
work_week=monday..friday;
day=1..31;
```

Обратите внимание на то, что при объявлении пользовательских типов между их именем и конструкцией, определяющей тип, ставится знак равенства.

После того, как тип объявлен, в разделе описания переменных можно пользоваться вновь введенным идентификатором.

Разделы описания констант, типов и переменных должны находиться перед основным блоком программы.

Таблица 2

Целочисленные типы		
Тип	**Диапазон возможных значений**	**Формат**
Shortint	-128..127	1 байт со знаком
Integer	-32768..32767	2 байта со знаком
Longint	-2147483648..2147483647	4 байта со знаком
Byte	0..255	1 байт без знака
Word	0..65535	2 байта без знака

Таблица 3

Вещественные типы			
Тип	**Диапазон возможных значений**	**Точность**	**Формат**
Real	2.9E-39..1.7E38	11—12 знаков	6 байт
Single	1.5E-45..3.4E38	7—8 знаков	4 байта
Double	5.0E-324..1.7E308	15—16 знаков	8 байт
Extended	3.4E-4932..1.1E4932	19—20 знаков	10 байт
Comp	-9.2E18..9.2E18	19—20 знаков	8 байт

Решение задач

1. Указать ошибки в следующих описаниях:

```
Const n=180; pi=3.1415;
Type sign=('a','b','c','d');
gl=(a, e, i, o, u);
sgl=(b..d, f, g);
log=Boolean;
sign='0'..'9';
```

2. Найти ошибки в следующей программе:

```
Program mistake_1;
Type month=(january, february, marth,
           april, may, june, july,
          august, september, october,
           november, december);
autumn=september..november;
Var m:autumn; d:'0'..'9'; k:0..9;
Begin
  Readln(m,d,k)
  If m>september Then d:=k
  Else k:=Ord(m)-8;
  Writeln(k,d+k);
End.
```

1.3.5. Преобразование типов. Совместимость типов

Повторение

1. Найти ошибки в следующей программе и объяснить, какие правила языка Паскаль в ней нарушены:

```
Program mistake_2;
Type month=(january,february,marth,
            april,may,june,july,august,
            september, october,
            november,december);
  winter=december..february;
  spring=marth..may;
  Var m:month; k:1..12;
  Begin
    Write('Введите месяц'); Readln(m);
    If m>spring Then m:=june;
    For k:=Ord(january) To Ord(m)
    Do m:=succ(m);
    Writeln(m);
  End.
```

На предыдущих занятиях были рассмотрены основные типы Паскаля и действия, которые можно совершать с переменными этих типов. Одним из самых распространенных действий является присваивание.

Рассмотрим такую ситуацию. Пусть заданы типы T1 и T2, а также описаны переменные p1 и p2 следующим образом:

```
Var     p1:T1; p2:T2;
```

Когда можно присвоить переменной p1 значения p2: p1:=p2? Чтобы ответить на этот вопрос, рассмотрим совместимость простых типов по присваиванию. Операция p1:=p2 является допустимой, если истинно одно из следующих утверждений:

- T1 и T2 — тождественные типы.

Типы являются *тождественными*, если они описаны одним и тем же идентификатором или происходят от одного и того же идентификатора.

Пример

```
Type T1 =Real; T2 =Real; T3=T1;
    T4=(red,green,blue,black,white);
    T5=(red,green,blue,black,white);
     T6=T4;
```

Здесь T1, T2 и T3 — тождественные типы, T4, T5 — не тождественные, поскольку (red, green, blue, black, white) не являются идентификаторами типа, T4, T6 являются тождественными.

- T2 является поддиапазоном типа T1.

Например, **Type** T1=Real; T2=Integer; (множество целых чисел входит в диапазон вещественных чисел).

- T1 и T2 являются отрезками одного и того же типа.

Например:

```
Type T1 = 1.. 100; T2 =  -3..20;
week=(d1,d2,d3,d4,d5,d6,d7);
working_week=(d1..d5);
```

Совместимость по присваиванию станет более понятна, если вспомнить, что переменные в памяти занимают определенное число байт. Так, переменная типа Integer занимает 2 байта, типа Real — 6 байт. Переменной, которая занимает 6 байт, можно присвоить значение, занимающее 2 байта, а наоборот — не всегда.

Совместимость типов необходима также в выражениях и операциях сравнения.

```
Program Example;
Var a:Byte; b:Integer; c:Longint;
Begin
  Writeln('Введите 2 числа:(Byte, Integer)');
  Readln(a, b);
  c:= a + b;
  Writeln(c);
End.
```

Задание

Исследовать, чему равно значение переменной c при различных значениях a и b.

Пусть a=100, b=32767. Чему равно значение c? При сложении этих чисел должно получиться число 32867, но, запустив программу, мы не получим этого числа.

Вычисление выражения осуществляется по следующим правилам: перед тем как выполнить операцию, оба операнда преобразуются к их общему типу. Общим типом является более сложный из типов двух операндов.

Задание

Определить тип следующих выражений: a+b; a+b+c; a+b+c+x, если переменные описаны так:

```
Var a:Byte; b:Integer; c:Longint;x:Real;
```

Выражение справа в операторе присваивания вычисляется независимо от размера или типа переменной слева.

Вернемся к примеру. Число 32 867 не входит в диапазон допустимых значений типа Integer, именно вследствие этого мы получаем неправильный результат. Чтобы этого не случилось, нужно преобразовать переменные к более "вместительному" типу.

Преобразование переменной к типу записывается следующим образом:

<идентификатор типа>(переменная).

Итак, чтобы получить правильный результат, нужно записать следующий оператор:

```
c:=Longint(a)+Longint(b);
```

или:

```
c:=a+Longint(b);
```

Задание

Можно ли записать `c:=Longint(a+b); Writeln(b*a*0.1); Writeln(0.1*a*b)`? Какой получится результат? Ответ поясните.

Для преобразования типов используются также следующие функции:

Round (от англ. *round* — круглый, округлять) — округление числа до ближайшего целого.

Trunc (от англ. *truncate* — урезать) — отбрасывание дробной части числа.

Int (от англ. *integer* — целый) — целая часть вещественного числа.

Frac (от англ. *fraction* — дробь) — дробная часть вещественного числа.

Low (от англ. *low* — низкий) — наименьшее значение данного порядкового или перечисляемого типа.

High (от англ. *high* — высокий, верх) — наибольшее значение данного порядкового или перечисляемого типа.

Пример

"Вечный календарь". Известно, что если дата лежит в диапазоне от 1582 до 4902 г., номер дня недели (номер воскресенья — 0, понедельника — 1, ..., субботы — 6) равен остатку от деления на 7 значения выражения

$$[2.6m-0.2]+d+y+[y/4]+[c/4]-2c \qquad (*)$$

Здесь d — номер дня в месяце (1, 2, ...); m — номер месяца в году, нумерация начинается с марта (у марта — номер 1, у апреля — номер 2, ..., у декабря — номер 10, январь и февраль считаются месяцами с номерами 11 и 12 предыдущего года); y — две младшие цифры года; c — две старшие цифры года; $[x]$ обозначает целую часть числа x.

Вычислить количество пятниц, приходящихся на 13-е число в XX столетии.

Вопросы для обсуждения

1. Какие понадобятся типы данных?

2. Какие величины остаются постоянными в процессе работы?

3. Какие переменные необходимы для решения задачи, каков их тип?

4. Для вычисления значения выражения (*) необходимо выполнить преобразование типов, для этого воспользуемся функцией Trunc, которая преобразует вещественное число в целое путем отбрасывания дробной части числа. Какие значения может принимать данное выражение?

5. Какие значения принимает функция Ord для констант типа month?

Решение

```
Program Example_26;
Type month=(marth, april, may, june,
           july, august, september,
           october, november, december,
           january, february);
     day=1..31;
     year=1582..4902;
     week=(sunday,monday,tuesday,
           wednesday,thursday,
           friday,saturday);
Const h=20;
      d: day=13;
      d_w: week= friday;
Var k:Integer;{для подсчета кол-ва пятниц}
   y: year; Mod_y:0..99; int_y:15..49;
    m:month;
   n:-50..1000;
Begin
  k:=0;
  For y:=(h-1)*100 To h*100-1 Do
  {просмотрим все годы столетия}
  For m:= marth To february
  Do{просмотрим все месяцы года}
  Begin
    Mod_y:=y mod 100;
    {найдем две последние цифры года}
    int_y:=y div 100;
    {найдем две первые цифры года}
   n:=trunc(2.6*(Ord(m)+1)0.2)+d+mod_y+
    trunc(mod_y/4)+ trunc(int_y/4)-2*int_y;
    If n mod 7=Ord(d_w) Then Inc(k);
  End;
  Writeln('количество пятниц, приходящихся
   на ',d,' число в ',h,' столетии равно ',k);
End.
```

При решении этой задачи нам понадобилось выполнить преобразование типов.

Пример

Найти k-е простое число в арифметической прогрессии 11, 21, 31, 41, 51, 61, ...

Решение

Для решения поставленной задачи необходимо просматривать числа последовательности и проверять каждое из них на простоту. Поскольку нам неизвестно, сколько членов последовательности необходимо просмотреть, мы должны просматривать последовательность до тех пор, пока не найдем k-е простое число. Для этого воспользуемся циклом с условием:

```
Program Example_27;
Var  k: Integer;
     n, p,d: Longint;
Begin
  Writeln('Введите номер числа');
  Readln(k);
  n:=0; p:=1;
  While n<k Do
  Begin
    Inc(p,10); d:=2;
   While (p mod d<>0) and (d<sqrt(p)) Do Inc(d);
    If d>=sqrt(p) Then Inc(n);
  End;
  Writeln(p);
  Readln;
End.
```

В этом решении мы смогли записать условие `d<sqrt(p)`, так как типы `Integer` и `Real` совместимы.

Решение задач

1. Даны натуральные числа, обозначающие число, месяц и год. Определить день недели, на который приходится указанная дата.

2. День учителя ежегодно отмечается в первое воскресенье октября. Дано натуральное число n, обозначающее номер года. Определить число, на которое приходится День учителя.

3. Рассмотрим некоторое натуральное число. Если это не палиндром, то изменим порядок его цифр на обратный и сложим исходное число с получившимся. Если сумма не палиндром, то над ней повторяется то же действие и т.д., пока не получится палиндром. Даны натуральные числа k, m, l ($k<l$). Проверить, верно ли, что для любого натурального числа из диапазона от k до l процесс завершается не позднее, чем после m таких действий.

4. Найти 100 первых простых чисел.

5. Дано натуральное число n, целые числа $a_1, a_2, ..., a_n$. Рассмотреть отрезки последовательности $a_1, a_2, ..., a_n$ (подпоследовательности идущих подряд членов), состоящие из:

a) полных квадратов;

b) степеней пятерки;

c) простых чисел.

В каждом случае получить наибольшую из длин рассматриваемых отрезков.

1.4. КОНТРОЛЬНЫЕ РАБОТЫ

1.4.1. Контрольная работа № 1

Вариант № 1

1. Дано натуральное число:

- найти сумму его цифр;
- верно ли, что число начинается и заканчивается одной и той же цифрой?

2. Найти все трехзначные числа, сумма цифр которых равна A, а само число делится на B (A и B вводятся с клавиатуры).

3. Дано натуральное число. Приписать к нему такое же число.

Вариант № 2

1. Дано натуральное число:

- найти произведение его цифр;
- верно ли, что в данном числе нет данной цифры A (A вводится с клавиатуры)?

2. Найти все трехзначные числа, которые при увеличении на 1 делятся на 2, при увеличении на 2 делятся на 3, при увеличении на 3 делятся на 4, а при увеличении на 4 делятся на 5.

3. Из данного натурального числа удалить все цифры A (A вводится с клавиатуры).

Вариант № 3

1. Дано натуральное число:

- найти количество цифр данного числа;
- верно ли, что данное число заканчивается на нечетную цифру?

2. Найти количество трехзначных чисел, сумма цифр которых равна A, а само число заканчивается цифрой B (A и B вводятся с клавиатуры).

3. Найти все симметричные натуральные числа (палиндромы) из промежутка от A до B (A и B вводятся с клавиатуры).

Вариант № 4

1. Дано натуральное число:

- найти количество четных цифр числа;
- верно ли, что данная цифра A встречается в числе более двух раз (A вводится с клавиатуры)?

2. Найти все четырехзначные числа, у которых сумма крайних цифр равна сумме средних цифр, а само число делится на 6 и 27.

3. Найти количество различных цифр данного натурального числа.

Вариант № 5

1. Дано натуральное число:

- найти первую и последнюю цифры числа;
- верно ли, что сумма цифр данного числа равна A (A вводится с клавиатуры)?

2. Найти все трехзначные числа, которые при делении на 2 дают остаток 1, при делении на 3 — остаток 2, при делении на 4 — остаток 3, а само число делится на 5.

3. Дано натуральное число. Приписать к нему такое же.

Вариант № 6

1. Дано натуральное число:

• сколько раз данная цифра *A* встречается в данном числе (*A* вводится с клавиатуры);

• верно ли, что в данном числе сумма цифр больше *B*, а само число делится на *B* (*B* вводится с клавиатуры)?

2. Найти все четырехзначные числа, в которых есть две одинаковые цифры.

3. Из данного натурального числа удалить все цифры *A* (*A* вводится с клавиатуры).

Вариант № 7

1. Дано натуральное число:

• найти вторую (с начала) цифру данного числа;

• верно ли, что данное число делится на *A*, *B* и *C* (*A*, *B* и *C* вводятся с клавиатуры)?

2. Найти все двузначные числа, которые при умножении на 2 заканчиваются на 8, а при умножении на 3 — на 4.

3. Найти количество различных цифр данного натурального числа.

Вариант № 8

1. Дано натуральное число:

• найти количество цифр данного числа, больших *A* (*A* вводится с клавиатуры);

• верно ли, что данное число принадлежит промежутку от *A* до *B* и кратно 3, 4 и 5 (*A* и *B* вводятся с клавиатуры)?

2. Найти сумму всех чисел из промежутка от *A* до *B*, кратных 13 и 5 (*A* и *B* вводятся с клавиатуры).

3. Найти все симметричные натуральные числа из промежутка от *A* до *B* (*A* и *B* вводятся с клавиатуры).

Вариант № 9

1. Дано натуральное число:

• сколько четных цифр в данном целом числе;

• верно ли, что в данном числе встречаются цифры *A* и *B* (*A* и *B* вводятся с клавиатуры)?

2. Найти все симметричные четырехзначные числа. Пример: 7667, 1331.

3. Дано натуральное число. Приписать к нему такое же.

Вариант № 10

1. Дано натуральное число:

• сколько раз первая цифра встречается в данном числе;

• верно ли, что данное число начинается на *A*, а заканчивается на *B* (цифры *A* и *B* вводятся с клавиатуры)?

2. Найти все четырехзначные числа, в которых ровно две одинаковые цифры.

3. Найти количество различных цифр данного натурального числа.

1.4.2. Контрольная работа № 2

Вариант № 1

1. Найти количество делителей натурального числа. Сколько из них четных?

2. Найти все натуральные числа *a, b* и *c* из интервала от 1 до 20, для которых выполняется равенство: $a^2+b^2=c^2$.

Вариант № 2

1. Найти сумму нечетных делителей натурального числа.

2. Найти все равновеликие прямоугольники, стороны которых выражены целыми числами *a* и *b*, а площадь равна *S* (*a* и *b* принадлежат интервалу от 1 до 20, а *S* вводится с клавиатуры).

Вариант № 3

1. Найти все натуральные числа из промежутка от 1 до 200, у которых количество делителей равно *N* (*N* вводится с клавиатуры).

2. Найти все натуральные числа *a, b* и *c* из интервала от 1 до 20, для которых выполняется равенство: $a+b^2=c^2$.

Вариант № 4

1. Найти все натуральные числа из промежутка от 1 до 200, у которых сумма делителей равна *S* (*S* вводится с клавиатуры).

2. Найти все такие тройки натуральных чисел *x, y* и *z* из интервала от 1 до 20, для которых выполняется равенство: $x^2-y=z^2$.

Вариант № 5

1. Найти количество делителей натурального числа, больших *K* (*K* вводится с клавиатуры).

2. Найти все натуральные числа *a, b* и *c* из интервала от 1 до 20, для которых выполняется равенство: $a^2b=c^2$.

Вариант № 6

1. Найти сумму целых чисел из промежутка от 1 до 200, у которых ровно 5 делителей.

2. Найти все такие тройки натуральных чисел *x, y* и *z* из интервала от 1 до 20, для которых выполняется равенство: $x^2+y^2=z^2$.

Вариант № 7

1. Найти все целые числа из промежутка от 100 до 300, у которых сумма делителей равна *K* (*K* вводится с клавиатуры).

2. Найти все такие тройки натуральных чисел *x, y* и *z* из интервала от 1 до 20, для которых выполняется равенство: $x^2+y^2-z^2=0$.

Вариант № 8

1. Найти все натуральные числа из промежутка от *a* до *b*, у которых количество делителей превышает заданное число *K*.

2. Найти все натуральные числа *a, b* и *c* из интервала от 1 до 20, для которых выполняется равенство: $a+b=c^2$.

Гл. редактор
С.Л.Островский
Зам. гл. редактора
Е.Б.Докшицкая
Редакция:
Н.Л.Беленькая,
Н.П.Медведева
Дизайн и компьютерная верстка:
Н.И.Пронская
Корректоры:
Е.Л.Володина,
С.М.Подберезина

121165, Киевская, 24
тел. 249 4896
Отдел рекламы
тел. 249 9870

ИНДЕКС ПОДПИСКИ
для индивидуальных подписчиков 32291
комплекта приложений 32744

Тел. (095)249 3138, 249 3386. Факс (095)249 3184

Объединение педагогических изданий "ПЕРВОЕ СЕНТЯБРЯ"
Регистрационный номер 012868
Отпечатано в типографии ОАО ПО "Пресса-1".
125865, ГСП, Москва, ул. Правды, 24.
Тираж 7000 экз.
Заказ №

Internet: inf@1september.ru
Fidonet: 2:5020/69.32
WWW: http://www.1september.ru