Что? Где? Когда? Чем? Как? Кого?

Кто?

Минимум


язык
програм-
мирования
алгоритми-
зация
информатика,
физика,
математика
интерес

Инструментарий.

Типы данных

Идентификатор	Длина (байт)	Диапазон значений	Операции
Целые типы			
integer	2	-32768..32767	+, -, /, *, Div, Mod, =, <=, =, <>, <, >
byte	1	0..255	+, -, /, *, Div, Mod, >=, <=, =, <>, <, >
word	2	0..65535	+, -, /, *, Div, Mod, >=, <=, =, <>, <, >
shortint	1	-128..127	+, -, /, *, Div, Mod, >=, <=, =, <>, <, >
longint	4	-2147483648..2147483647	+, -, /, *, Div, Mod, >=, <=, =, <>, <, >
Вещественные типы			
real	6	2,9x10-39 - 1,7x1038	+, -, /, *, >=, <=, =, <>, <, >
single	4	1,5x10-45 - 3,4x1038	+, -, /, *, >=, <=, =, <>, <, >
double	8	5x10-324 - 1,7x10308	+, -, /, *, >=, <=, =, <>, <, >
extended	10	3,4x10-4932 - 1,1x104932	+, -, /, *, >=, <=, =, <>, <, >
Логический тип			
boolean	1	true, false	Not, And, Or, Xor, =, <=, =, <>, <, >
Символьный и строковый тип			
char	1	все символы кода ASCII	+, >=, <=, =, <>, <, >
string	255	все символы кода ASCII	=, <>, <, >

Инструментарий.

Синтаксис описания констант:

const

ИмяКонстанты= <значение>;

```
const  Max = 100;
```

Описание переменных

VAR

ИмяПеременной : ИмяТипа;

или

VAR

ИмяПеременной1,

ИмяПеременной2,

ИмяПеременной3 : ИмяТипа;

```
VAR
    I : integer;
    r1, r2: real;
```

Инструментарий.

Ввод

read (<список_ввода>);

readln (<список_ввода>);

```
read(c);
readln(k, x, c, s);
```

Вывод

write(<список_вывода>) ;
writeln(<список_вывода>) ;

```
writeln(a, b, c);
writeln('Молодец!');
writeln('Сумма равна ', sum);
writeln(X:4:2);
```

Инструментарий.

Условия

if условие **then** оператор;

if условие **then** оператор **else** оператор;

if логическое выражение **then**

begin

оператор1;

оператор2;

end

else

begin

оператор1;

оператор2;

end;

Инструментарий.

Операторы отношения Паскаля:

больше **>**, меньше **<**,
больше или равно **>=**,
меньше либо равно **<=**,
сравнение **=**, не равно **<>**

Логические операции

AND *(И),* ***OR*** *(ИЛИ) ,*
XOR *(исключающее или),* ***NOT*** *(не)*

Сложные условия

```
if (x >= 25) and (x <= 40) then
        writeln (‘лежит')
else
        writeln ('Не лежит');
```

Инструментарий.

Математические функции

Функция	Назначение
ABS(x)	Вычисление абсолютного значения x: |x|
SQR(x)	Вычисление квадрата x: x*x
SQRT(x)	Вычисление квадратного корня из x
COS(x), SIN(x)	Вычисление косинуса x: cos x Вычисление синуса x: sin x
Power(x,n)	Вычисление степени n для числа x
A DIV B	Вычисление частного при делении A на B с отбрасыванием остатка
A MOD B	Нахождение остатка от делении A на B
TRUNC(x)	Нахождение целой части x
RANDOM(x)	Псевдослучайное число в интервале [0, x]
ROUND(x)	Округление значения x в сторону ближайшего целого
FRAC(X)	Возвращает дробную часть x
INT(X)	Возвращает целую часть x
Pi	Значение математической постоянной π

Инструментарий.

Работа со строкой (символом)

<>, >, <, +, **S[i]**

Строковые функции

Concat (s1, s2, ..., sn)	Сложение строк
Copy (s, start, len)	Получение подстроки, начиная с позиции start, длиной len
Delete (s, start, len)	Удаление из строки, начиная с позиции start, длиной len
Insert (subs, s, start)	Вставляет в строку s подстроку subs, начиная с позиции start.
Length (s)	Возвращает фактическую длину строки s
Pos (subs, s)	Ищет первое вхождение подстроки subs в строку s
Str (x, s)	преобразует числовое значение x в строку s
Val (s, x, errcode)	преобразует строку s в значение числовой переменной x

Символьные функции

Ord (c)	возвращает числовой код символа из таблицы ASCII
Chr (i)	Возвращает символ из таблицы ASCII по коду
Pred(c)	значение предыдущего символа из таблицы ASCII
Succ(c)	значение следующего символа из таблицы ASCII

Инструментарий.

Таблица ASCII

Dec	Char	Dec	Char	Dec	Char	Dec	Char	Dec	Char	Dec	Char
0	NUL	21	NAK	42	*	63	?	84	T	105	i
1	SOH	22	SYN	43	+	64	@	85	U	106	j
2	STX	23	ETB	44	,	65	A	86	V	107	k
3	ETX	24	CAN	45	-	66	B	87	W	108	l
4	EOT	25	EM	46	.	67	C	88	X	109	m
5	ENQ	26	SUB	47	/	68	D	89	Y	110	n
6	ACK	27	ESC	48	0	69	E	90	Z	111	o
7	BEL	28	FS	49	1	70	F	91	[	112	p
8	BS	29	GS	50	2	71	G	92		113	q
9	TAB	30	RS	51	3	72	H	93	]	114	r
10	LF	31	US	52	4	73	I	94	^	115	s
11	VT	32	(sp)	53	5	74	J	95	_	116	t
12	FF	33	!	54	6	75	K	96	`	117	u
13	CR	34	"	55	7	76	L	97	a	118	v
14	SO	35	#	56	8	77	M	98	b	119	w
15	SI	36	$	57	9	78	N	99	c	120	x
16	DLE	37	%	58	:	79	O	100	d	121	y
17	DC1	38	&	59	;	80	P	101	e	122	z
18	DC2	39	'	60	<	81	Q	102	f	123	{
19	DC3	40	(	61	=	82	R	103	g	124	|
20	DC4	41	)	62	>	83	S	104	h	125	}

Инструментарий.

Множества

Типы **byte (0..255)** и **char (0..255)**

```
var
    ch: set of char;
    ch1: set of 'A'..'Z';
Begin
    ch:=['C', 'E'..'M', 'Z'];
```

Операции

+, *, -, include, exclude

<=, >=, =, <>

in

Инструментарий.

Цикл со счетчиком

for *счетчик:=значение* ***to (downto)*** *конечное_значение* ***do*** *тело_цикла;*

```
for i:=1 to n do
begin
        K:= k+k*i;
        Writeln(k);
end;
```

Цикл с предусловием

while условие **do** тело_цикла;

```
while K<100 do begin
        i:=i+1;
        K:= k+k*i;
end;
Writeln(i);
```

Инструментарий.

Массивы

Одномерные

```
var a: array [1..3] of integer;
begin
        a[1]:=500;
```

Двумерные

```
var X:array [1..5, 1..5] of integer;
begin
        a[1,5]:=2;
        Write a[1,5];
```

Заполнение случайными числами

```
randomize;
for i:=1 to N do
    for j:=1 to M do
        x [I, J]:=random (100);
```

Инструментарий.

Объявление файловой переменки

f: text;

Функции работы с файлами

Assign (f, ‘путь’)	устанавливает связь между файловой переменной и физическим именем
Reset (f)	открывает существующий файл для чтения.
Rewrite(f)	Открывает/создает файл для записи (если файл ранее существовал, вся предыдущая информация из него стирается)
Readln (f , st)	чтение строки st из файла f и переход на начало следующей строки
Writeln (f, st)	запись строки st в файл f и маркера конца строки
Append (f)	процедура, открывающая файл f для добавления строк в конец файла
Eoln (st)	логическая функция, результат выполнения которой равен TRUE, если достигнут маркер конца строки st

Крайние цифры (20 баллов).

Дано натуральное число N (1<=N<=10000).

Требуется написать программу, определяющую для N, какая цифра больше – первая или последняя.

Технические требования:

Входной файл: INPUT.TXT

Выходной файл: OUTPUT.TXT

Ограничение по времени: 1 секунда.

Формат входных данных:

В единственной строке задано число N.

Формат выходных данных:

В единственной строке необходимо вывести один символ: «=», если цифры одинаковые, «<», если первая цифра меньше последней, и «>», если первая цифра больше последней.

Пример файла входных данных:

1234

Пример файла выходных данных:

<

Крайние цифры (20 баллов).

---

1. Считать данные из файла
2. Выделить последнюю и первую цифру
    1. Последняя цифра - взять остаток от деления на 10
    2. Первая – делить 10 пока результат деления больше 10
3. Сравнить цифры
    1. если они равны вывести =
    2. если первая больше вывести >
    3. если первая меньше вывести <

Крайние цифры (20 баллов).

```
var
  n, first:longint;
  f:text;
begin
    assign(f,'input.txt');
    reset(f);
    readln(f,n);
    close(f);
    first:=n;
    while first>10 do  first:=first div 10;
    n:=n mod 10;
    assign(f,'output.txt');
    rewrite(f);
    if first=n then write(f,'=')
    else  if first<n then write(output,'<')
    else write(f,'>');
    close(f);
end.
```

Крайние цифры (20 баллов).

Найдем ошибки:

```
var
  n,first:longint;
begin
  assign(input,'input.txt');
  assign(output,'output.txt');
  read(n);
  first:=n;
  while first>10 do first:=first div 10;
  if first=n mod 10  then  write('=');
  if first=n mod 10  then  write('<');
  if first=n mod 10  then  write('>');
  close(input);
  close(output);
end.
```

Отрезки (30 баллов)

Дано натуральное число N (1<=N<=100). Даны координаты вершин N отрезков.

Требуется написать программу, определяющую количество отрезков, пересекающих ось абсцисс. При этом будем считать отрезок пересекающим ось абсцисс, если отрезок имеет на оси ровно одну точку, не являющуюся вершиной.

Технические требования:

Входной файл: INPUT.TXT

Выходной файл: OUTPUT.TXT

Ограничение по времени: 1 секунда.

Формат входных данных:

В первой строке задано число N. В следующих N строках заданы координаты вершин отрезков так, что параметры каждого отрезка содержатся в отдельной строке: абсцисса первой вершины, ордината первой вершины, абсцисса второй вершины, ордината второй вершины. Координаты точек являются целыми числами, по абсолютной величине не превосходящими 1000.

Формат выходных данных:

В единственной строке выводится одно число – количество отрезков, пересекающих ось абсцисс.

Пример входных данных:	выходных данных:
2	1
0 0 1 1	
3 3 8 –1	

Отрезки (30 баллов)

```
var
  n,s,i:longint;
  x1,y1,x2,y2:longint;
  f:text;
begin
        assign(f,'input.txt');
        reset(f);
        readln(f,n);
        s:=0;
        for i:=1 to n do
          begin
                readln(f,x1,y1,x2,y2);
                if y1*y2<0 then s:=s+1;
          end;
         close(f);
         assign(f,'output.txt');
         rewrite(f);
         write(f,s);
         close(f);
end.
```

Красное словцо (50 баллов)

Требуется написать программу, которая печатает слово, получающееся из исходного путем замены строчных букв на прописные, а прописных – на строчные. Будем называть словом любую последовательность букв латинского алфавита, длиной не более 80 символов.

Технические требования:

Входной файл: INPUT.TXT

Выходной файл: OUTPUT.TXT

Ограничение времени: 2 секунды.

Формат входных данных:

В единственной строке задано исходное слово.

Формат выходных данных:

В единственной строке должно содержаться слово, получающееся из исходного путем замены строчных букв на прописные, а прописных на строчные.

Пример файла входных данных:

PCProgram

Пример файла выходных данных:

PcpROGRAM

Красное словцо (50 баллов)

```
var
  s:string;
  i:longint;
begin
  assign(input,'input.txt');
  assign(output,'output.txt');
  reset(input);
  rewrite(output);
  readln(s);
  for i:=1 to length(s) do
    if s[i] in ['a'..'z']
      then
        s[i]:=chr(ord(s[i])-32)
      else
        s[i]:=chr(ord(s[i])+32);
  writeln(s);
  close(input);
  close(output);
end.
```

Робот (50 баллов)

В точке отсчета координатной плоскости находится робот, который может делать шаги в четырех направлениях: вверх, вниз, влево и вправо. При этом одна из координат изменяется на единицу – робот не может оставаться в той же точке, в которой он находился до выполнения шага. Известно количество шагов N, которые может сделать робот, пока у него не кончится заряд батарей.

Требуется написать программу, определяющую количество точек, которые достижимы для робота при заданных ограничениях.

Технические требования:

Входной файл: INPUT.TXT

Выходной файл: OUTPUT.TXT

Ограничение времени: 1 секунда.

Формат входных данных:

В единственной строке задано целое неотрицательное число N (1<=N<=10000).

Формат выходных данных:

В единственной строке необходимо вывести количество точек, которые достижимы для робота при заданной величине заряда.

Пример файла входных данных:

1

Пример файла выходных данных:

4

Робот (50 баллов)

```
var
  n,i,s:longint;
  f: text;
begin
  assign(f,'in.txt');
  reset(f);
  read(f,n);
  close(f);
  s:=1;
  for i:=1 to n do s:=s+4*i;
  assign(f,'output.txt');
  rewrite(f);
  write(f,s);
  close(f);
end.
```

Игра (50 баллов)

Вася и Петя играют в следующую игру. На доске написано натуральное число, состоящее из N цифр (1<=N<=100). Ребята по очереди выбирают по одной цифре, причем они могут выбрать одну и ту же цифру. Выбранные числа ребята перемножают и записывают произведение в тетрадь. Игра продолжается, пока все возможные произведения (их будет N*N) не появятся в тетради.

Требуется написать программу, которая определяет, сумму всех чисел, которые Вася и Петя запишут в тетрадь.

Технические требования:

Входной файл: INPUT.TXT

Выходной файл: OUTPUT.TXT

Ограничение времени: 1 секунда.

Формат входных данных:

В единственной строке задано натуральное число K ($1<=K<10^{100}$).

Формат выходных данных:

В единственной строке вывести сумму всех попарных произведений цифр.

Пример файла входных данных:

12

Пример файла выходных данных:

9

Игра (50 баллов)

```
var
  i,res:longint;
  s:string;
begin
  assign(input,'input.txt');
  assign(output,'output.txt');
  readln(s);
  for i:=1 to length(s) do
res:=res+ord(s[i])-48;
  res:=res*res;
  write(res);
  close(input);
  close(output);
end.
```

Игра (50 баллов)

```
var
  i,j,res:longint;
  s:string;
begin
  assign(input,'input.txt');
  assign(output,'output.txt');
  readln(s);
  for i:=1 to length(s) do
    for j:=1 to length(s) do
res:=res+(ord(s[i])-48)*(ord(s[j])-48);
  write(res);
  close(input);
  close(output);
end.
```

Проверка времяни

procedure GetTime(var Hour, Minute, Second, Second100:word);

```
uses dos;
var
  h1,m1,s1,t1:word;
  h2,m2,s2,t2:word;
  d:longint;
begin
  gettime(h1,m1,s1,t1);
  ... {любое действие, продолжительность которого измеряем}
  gettime(h2,m2,s2,t2);
  {вычислим время выполнения d - результат будет в сотых долях секунды}
  d:=(longint(h2)*360000+longint(m2)*6000+s2*100+t2)-
{количество сотых долей секунды после выполнения}
     (longint(h1)*360000+longint(m1)*6000+s1*100+t1); {их количество до выполнения действия}
  writeln('Действие выполнялось ',d/100:0:2,' секунды');
end.
```

Полезности

Обучение в форме игр

CodeCombat. Это RPG, в которой вам предстоит проходить уровни и прокачивать героя, параллельно с этим изучая один из доступных языков программирования. Первые уровни, рассчитанные на 1-3 часа времени (в зависимости от ваших способностей), доступны бесплатно, и позволят вам понять, нравится ли вам вообще это занятие.

JavaRush - онлайн-игра, нацеленная на обучение вас программированию на Java. Тут всё уже несколько сложнее, уровней доступно больше, к каждому имеется лекция.

Для тех, кто не готов к знакомству со взрослыми языками программирования (или уже знаком, и хочет просто поиграться), имеется очаровательная и бесплатная игра Cargo-Bot, доступная под iPad. Там можно познакомитесь с построением алгоритмов и решите ряд довольно интересных задач.

Информация взята с сайта: Уютный чердачок информатики
http://inform-attic.ru/article/samoobuchenie-programmirovaniyu