Циклы

Имя, фамилия ученика	Класс	Дата

1. Счётчик цикла (1 Б.)

Дано: $for\ i := 1\ to\ 11\ do$

При первом выполнении тела цикла $i =$ ☐ .

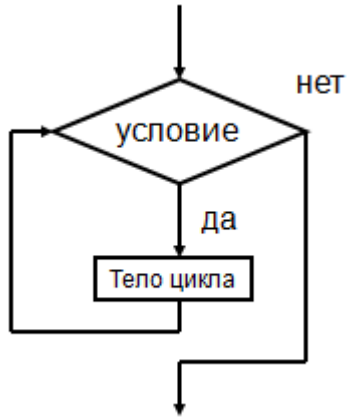
2. Оператор for (1 Б.)

Сколько раз будет выполнен цикл $for\ n := 1\ to\ 30\ do$?

Ответ: .....

3. Циклические алгоритмы (1 Б.)

Выбери верный ответ.



- ☐ цикл с неизвестной величиной
- ☐ цикл с предусловием
- ☐ цикл с параметром по убывающим значениям параметра
- ☐ цилиндрический цикл

4. Числа на экране в обратном порядке (3 Б.)

Запиши программу на языке Паскаль.
Вывести на экран числа от 5 до 25 в обратном порядке.

var

```
i: integer;
begin
for i:= _____ downto _____ do
        ☐ i
        ☐ 25
        ☐ 'i'
writeln(☐ 5 );

readln;
end.
```

5. Различные варианты программирования циклического алгоритма (0 Б.)

Особенностью программирования является то, что для решения одной и той же задачи могут быть созданы разные программы. Вы могли убедиться в этом, программируя ветвления. Рассмотрим пример, показывающий, что и циклический алгоритм может быть запрограммирован разными способами.

Пример:

Напишем программу, в которой осуществляется ввод целых чисел (ввод осуществляется до тех пор, пока не будет введён ноль) и подсчёт количества введённых положительных и отрицательных чисел.
*Так как здесь в явном виде задано условие окончания работы, то воспользуемся оператором **repeat**.*

```
var n, k1, k2: integer;
begin
  k1:=0;
  k2:=0;
  repeat
    write ('Введите целое число>>');
    readln (n);
    if n>0 then k1:=k1+1;
    if n<0 then k2:=k2+1;
  until n=0;
  writeln ('Введено:');
  writeln ('положительных чисел - ', k1);
  writeln ('отрицательных чисел - ', k2)
end.
```

*Имеющееся условие окончания работы можно достаточно просто преобразовать в условие продолжения работы — работа продолжается, пока $n \neq 0$. И мы можем воспользоваться оператором **while**.*

```
  var n, k1, k2: integer;
begin
  k1:=0;
  k2:=0;
  while n<>0 do
  begin
    writeln('Введите целое число>>');
    read (n);
    if n>0 then k1:=k1+1;
    if n<0 then k2:=k2+1;
  end;
  writeln('Введено:');
  writeln('положительных-', k1);
  writeln ('отрицательных-', k2)
end.
```

Источники:
Босова Л. Л., Босова А. Ю., Информатика: учебник для 8 класса. М. : БИНОМ. Лаборатория знаний, 139 с.

6. Условный оператор (1 Б.)

Является ли условным оператором следующая последовательность символов?

$if\ x < y < z;\ then\ a := 149$

- ☐ не является
- ☐ является

7. Программа (4 Б.)

Дана программа на языке Паскаль.
Программа определения весовой категории в зависимости от веса спортсмена. Если вес спортсмена меньше 58, то «лёгкая категория» иначе «средняя категория». Заполните пустые окошки.

```
program ves;
const
A1='легкая категория';
A2='средняя категория';
var
b : integer;
begin
write('введите вес спортсмена .....= ');
readln(.....);
if .....< ..... then writeln(A1) else writeln(A2);
readln;
end.
```

8. Задача про число (2 Б.)

Дано целое число. Если оно является положительным, то прибавить к нему 57; в противном случае не изменять его. Вывести полученное число.

```
program cislo;
var
a: integer;
begin
writeln('Введи число a');
readln(a);
if a < 0 then a:=a+57;
writeln(a);
end.
```

```
program cislo;
var
a: integer;
begin
writeln('Введи число a');
readln(a);
if a > 0 then a:=a;
writeln(a);
end.
```

```
program cislo;
var
a: integer;
begin
writeln('Введи число a');
readln(a);
if a > 0 then a:=a+57;
writeln(a);
end.
```

9. Условный оператор (0 Б.)

Условный оператор — это конструкция языка программирования, которая обеспечивает выполнение одного из двух наборов команд в зависимости от значения некоторого условия.

Общий вид:

```
if <условие> then <оператор_1> else
                <оператор_2>
```

Для записи неполных ветвлений используется неполная форма условного оператора:

```
if <условие> then <оператор>
```

Слова **if — then — else** переводятся с английского языка на русский как **если — то — иначе**, что полностью соответствует записи ветвления на алгоритмическом языке.

Обрати внимание!

Перед **else** знак «;» не ставится.

В качестве условий используются логические выражения:

- простые — записанные с помощью операций отношения;
- сложные — записанные с помощью логических операций.

Пример:

Алгоритм определения принадлежности точки x отрезку $[a, b]$. Если точка x принадлежит данному отрезку, то выводится ответ ДА, в противном случае — НЕТ. Запишем на языке Паскаль алгоритм определения принадлежности точки x отрезку $[a, b]$.

```
program uslovie;
  var x, a, b: real;
begin
  writeln ('Определение принадлежности
           точки отрезку');
  write ('Введите a, b>>');
  readln (a, b);
  write ('Введите x>>');
  readln (x);
  if (x>=a) and (x<=b) then
  writeln ('Точка принадлежит отрезку')
  else writeln ('Точка не принадлежит отрезку')
end.
```

Пример:

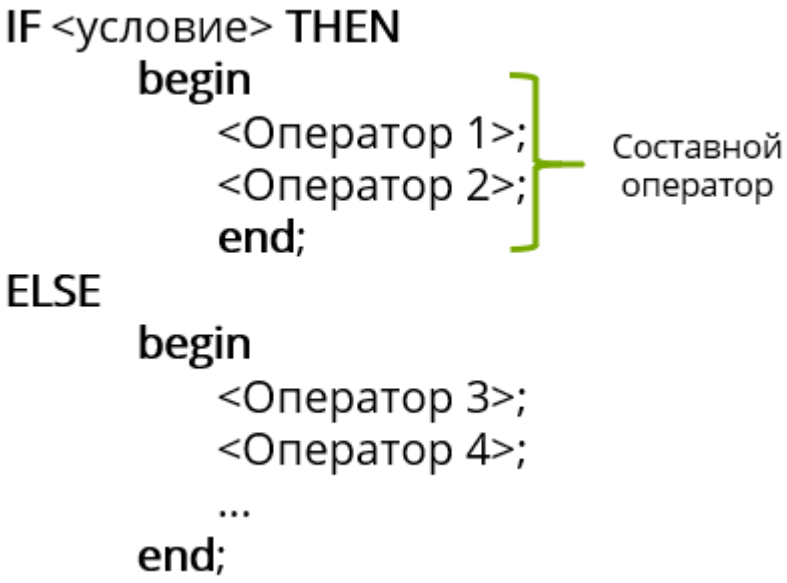
Алгоритм, в котором переменной y присваивается значение большей из трёх величин a, b и c. Запишем на языке Паскаль данный алгоритм.

```
program tocka;
  var y, a, b, c: integer;
begin
  writeln ('Нахождение наибольшей из трёх величин');
  write ('Введите a, b, c>>');
  readln (a, b, c);
  y:=a;
  if (b>y) then y:=b;
  if (c>y) then y:=c;
  writeln ('y=', y)
end.
```

Источники:
Босова Л. Л., Босова А. Ю., Информатика: учебник для 8 класса. М. : БИНОМ. Лаборатория знаний, 129 с.

10. Составной оператор (0 Б.)

Условный оператор не позволяет выполнять несколько действий между **then** и **else**.
В этом случае используем составной оператор. Он позволяет выполнять последовательность из нескольких действий после **then**. Тогда эти действия записываются между служебными словами **begin** и **end**.



Рассмотрим программу, которая решает любое квадратное уравнение.

```
program kv_uravnenie;
  var a, b, c: real;
  var d: real;
  var x, x1, x2: real;
begin
  writeln ('Решение квадратного уравнения');
  write ('Введите коэффициенты a, b, c  >>');
  readln (a, b, c);
```

```
  d:=b*b-4*a*c;
  if d<0 then writeln ('Корней нет');
  if d=0 then
begin
       x:=-b/2*a;
       writeln ('Корень уравнения x=', x:9:3)
     end;
if d>0 then
begin
      x1:=(-b+sqrt(d))/(2*a);
      x2:=(-b-sqrt(d))/(2*a);
      writeln ('Корни уравнения:');
      writeln ('x1=', x1:9:3);
      writeln ('x2=', x2:9:3)
     end;
end.
```

11. Математическое выражение (1 Б.)

Запиши математическое выражение $(15+z)^2$ на языке Паскаль. *(При записи ответа используй прописные буквы).*

Ответ: ☐ (☐ ☐ z)

12. Выражение на языке Паскаль (1 Б.)

Выражение на языке Паскаль: $sqrt(sqr(z)-10*z+6)$.
Данное выражение на математическом языке:

- ☐ $\left(z^2-10z+6\right)^2$
- ☐ $\sqrt[5]{z^2-10z+6}$
- ☐ $\sqrt{z^2-10z+6}$
- ☐ $\left|z^2-10z+6\right|$

13. Длина гипотенузы (2 Б.)

Выбери программу, которая вычисляет длину гипотенузы.

```
var
m,y,h:real;
begin
writeln('Введи длины катетов m и y ');
readln(m,y);
h:=m*m+y*y;
writeln('h=',c:8:2);
readln;
end.
```

```
var
m,y,h:real;
begin
writeln('Введи длины катетов m и y ');
readln(m,y);
h:=sqrt(m*m+y*y);
writeln('h=',h:8:2);
readln;
end.
```

```
var
m,y,h:real;
begin
writeln('Введи длины катетов m и y ');
readln(m,y);
h:=sqr(sqrt(m)+sqrt(y));
writeln('h=',c:8:2);
readln;
end.
```

14. Числовые типы данных (0 Б.)

Вы уже знакомы с основными числовыми типами данных **integer** и **real**. К ним применимы стандартные функции, часть из которых приведена в таблице.

Стандартные функции Паскаля

Функция	Назначение	Тип аргумента	Тип результата
abs (x)	Модуль x	integer, real	Такой же, как у аргумента
sqr (x)	Квадрат x	integer, real	Такой же, как у аргумента
sqrt (x)	Квадратный корень из x	integer, real	real
round (x)	Округление x до ближайшего целого	real	
frac (x)	Дробная часть x	real	
int (x)	Целая часть x	real	
random	Случайное число от 0 до 1	-	real
random (x)	Случайное число от 0 до x	integer	integer

Исследуем работу функций **round**,**int** и **frac**, применив их к некоторому вещественному x. Соответствующая программа будет иметь вид:

```
program n_3;
 var x: real;
begin
  writeln ('Исследование функций round, int, frac');
  write ('Введите x>>');
  readln (x);
  writeln ('Округление - ', round(x));
  writeln ('Целая часть - ', int(x));
  writeln ('Дробная часть - ', frac(x))
end.
```

Источники:
Босова Л. Л., Босова А. Ю., Информатика: учебник для 8 класса. М. : БИНОМ. Лаборатория знаний, 120 с.

15. Символьный и строковый типы данных (0 Б.)

Значением символьной величины (тип **char**) в языке Паскаль является любой из символов, который можно получить на экране нажатием на клавиатуре одной из клавиш или комбинации клавиш, а также некоторых других символов, в том числе и невидимых. Каждому такому элементу присваивается код — число 0 до 255.

Первым 32 кодам соответствуют управляющие символы, а остальные коды изображаемым символам. К изображаемым символам относится и пробел, имеющий код 32.

Кодам от 33 до 127 соответствуют знаки препинания, знаки арифметических операций, цифры, прописные и строчные латинские буквы. Буквам национального алфавита соответствуют коды с номерами 128 и далее.
В тексте программы константу символьного типа можно задать, заключив любой изображаемый символ в апострофы: **'5' , 'B'**.

Если значение символьной переменной считывается с клавиатуры, то его следует набирать без апострофов. Чтобы найти код символа, используют функцию ord, где в качестве параметра задают символ.

Чтобы по коду узнать символ, используют функцию chr, где в качестве параметра указывают код символа.

Значением строковой величины (тип **string**) является произвольная последовательность символов, заключенная в апострофы. В Паскале (как и в алгоритмическом языке) строки можно сцеплять.
Запишем на языке Паскаль программу, в которой для введённой с клавиатуры буквы на экран выводится её код. Затем на экран выводится строка, представляющая собой последовательность из трёх букв используемой кодовой таблицы: буквы, предшествующей исходной; исходной буквы; буквы, следующей за исходной.

```
program n_5;
 var a: char; kod: integer; b: string;
begin
 writeln ('Код и строка');
 write ('Введите исходную букву>>');
 readln (a);
 kod:=ord(a);
 b:=chr(kod–1)+a+chr(kod+1);
 writeln ('Код буквы ', a, '-', kod);
 writeln ('Строка: ', b)
end.
```

16. Логический тип данных (0 Б.)

Логические величины могут принимать два значения: 0 и 1. В Паскале это $false$ и $true$. Константы определены таким образом, что **false < true**. В результате выполнения операций сравнения числовых, символьных, строковых и логических выражений получаются логические значения. Следовательно, в Паскале логической переменной можно присваивать результат операции сравнения.

Пример:

Составим программу, которая определяет истинность высказывания «Число n является чётным» для произвольного целого числа n.
Пусть ans — логическая переменная, а n — целая переменная.
Тогда в результате выполнения оператора присваивания
ans:=n mod 2 = 0
переменной ans будет присвоено значение $true$ при любом чётном n и $false$ в противном случае.

```
program n_6;

 var n: integer; ans: boolean;

begin

 writeln ('Определение истинности высказывания о чётности числа');

 write ('Введите исходное число>>');

 readln (n);

 ans:=n mod 2=0;

 writeln ('Число ', n,' является четным - ', ans)

end.
```

Логическим переменным можно присваивать значения логических выражений, построенных с помощью известных вам логических функций **и, или, не**, которые в Паскале обозначаются соответственно **and, or, not**.

Пример:

Напишем программу, определяющую истинность высказывания «Треугольник с длинами сторон a, b, c является равнобедренным» для произвольных целых чисел a, b, c.

```
program n_7;
 var a, b, c: integer; ans: boolean;
begin
 writeln ('Определение истинности высказывания
                  о равнобедренном треугольнике');
 write ('Введите значения a, b, c>>');
 readln (a, b, c);
 ans:=(a=b) or (a=c) or (b=c);
 writeln ('Треугольник с длинами сторон ', a, ',', b,
              ',', c, ' является равнобедренным - ', ans)
end.
```

Источники:
Босова Л. Л., Босова А. Ю., Информатика: учебник для 8 класса. М. : БИНОМ. Лаборатория знаний, 123 с.

1