Ливенский филиал ПГУ

Кафедра естественнонаучных дисциплин

Шатохина Елена Николаевна

ОПОРНЫЕ КОНСПЕКТЫ
ЛИСТИНГ ПРОГРАММ

по дисциплине

ОСНОВЫ ПРОГРАММИРОВАНИЯ

2015

«Алгебра логики». Опорный конспект

Высказывание – утверждение, про которое можно точно сказать, является ли оно ложным или истинным.

Алгебра логики – раздел математики, в которой изучаются логические операции над высказываниями.

Значение высказывания

Нет	Да
Ложь	Истина
False	True
0	1

Составное высказывание – логическое высказывание состоящее из двух и более высказываний.

Базовые операции над высказываниями

Конъюнкция	**Дизъюнкция**	**Отрицание**
A ^ B	A V B	$\overline{A}$
И	**ИЛИ**	**НЕ**

«Сложные высказывания». Опорный конспект

A = B ^ (CvD) B = 1,C = 0, D = 1, A = ?

A = B ^ (CvD) = 1 ^ (0 v 1) = 1 ^ 1 = 1

A = (Bv$\overline{C}$) ^ ($\overline{DvC}$)B= 0, C = 0, D= 1, A = ?

A = (B v $\overline{C}$) ^ ($\overline{D \text{ v } C}$) = (0 v $\overline{0}$) ^ ($\overline{1 \text{ v } 0}$) = (0 v 1) ^$\overline{1}$ = 1 ^ 0 = 0

«Язык программирования Паскаль». Опорный конспект

Никлаус Вирт – швейцарский учёный, автор языка программирования Паскаль.
Блез Паскаль –французский писатель, математик, физик, механик,изобретатель, философ. Создатель первого арифмометра в Европе – счётной машины «паскалина». В честь Блеза Паскаля Никлаус Вирт назвал свой новый язык программирования.

Основные черты языка Паскаль как средства обучения программированию

- структурированность;
- академичность;
- ясный визуальный стиль;
- максимальная приближенность к английскому языку.

Интегрированные среды разработки, использующие диалекты языка Паскаль

- Delphi
- Lazarus
- MSEide+MSEgui
- PascalABC.NET

Известные компиляторы
BorlandTurbo Pascal, Free Pascal, GNU Pascal, Microsoft Pascal, Object Pascal (Apple) , Virtual Pascal

НекоторыепрограммынаписанныенадиалектахПаскаля
TotalCommander, Skype, PL/SQLDeveloper, TOAD, PHPEdit, BSPlayer,MacromediaHomeSite, AdAware, FLStudio, AgeofWonders, Doom: TheRoguelike, AuslogicsDiskDefrag, AIMP2/AIMP3, Spybot, The KMPlayer, PE Explorer, QIP, Photoshop 1.0имногиедругие…

«Общая структура программ». Опорный конспект

Общая структура программ

programname_of_program;

{Разделописаний}

begin

(* Тело программы *)

end.

Служебные слова

and array as begin case class const constructor destructor div do downto else end event except file final finalization finally for foreach function goto if implementation in inherited initialization interface is label lock mod nil not of operator or procedure program property raise record repeat set shlshrsizeof template then to try type typeof until uses using var where while with xor

Подразделы раздела описаний

Обозначение	Служебное слово
Метки	label
Типы	type
Переменные	var
Константы	const
Процедуры	procedure
Функции	function

Некоторые дополнительные модули

Название	Назначение
GraphABC	Растровая графика
Arrays	Массивы
ABCSprites	Анимация
Timer	Время
Sounds	Звуки
Events	События

Типы переменных

Обозначение	Тип
Целые числа	integer, byte
Вещественные числа	real
Символы	char
Строки	string
Массивы	array
Логические переменные	boolean

Комментарии

(* Способ 1: комментарий внутри обычных скобок со знаком умножения *)
{Способ 2: комментарий внутри фигурных скобок }
(* { (* Пример того как одни комментарии *) можно вставлять } внутри других комментариев *)

«Линейные программы»
Листинг 1. «Сфера»

Задание:

Задано целочисленное число, означающее радиус сферы. Найти её площадь и объём.

Вводные данные:радиус сферы R.

Результирующие данные:площадь сферы S, объём сферы V.

Листинг:

```
program sphere;

var
   R : integer;
   S, V : real;

const
    Pi = 3.1415926;

begin

{Получение радиуса}
write('Введите радиус сферы: ');
readln(R);

    {Вычисляем площадь и объём}
S := 4 * Pi * sqr(R);
    V := 4 / 3 * (Pi * sqr(R) * R);

{Выводим результаты на экран}
writeln('Площадь сферы: ', S:1:3, ' м2');
writeln('Объём сферы: ', V:1:3, ' м3');

end.
```

Тестирование программы:

Введите радиус сферы: 5 Площадь сферы: 314.159 м2 Объём сферы: 523.599 м3	Введите радиус сферы: 7 Площадь сферы: 615.752 м2 Объём сферы: 1436.755 м3

«Линейные программы». Опорный конспект

```
program name_of_program;

        {Раздел описаний}

const
        {Описание констант}

var
        {Описание переменных}

begin

        (* Команда 1 *)
        (* Команда 2 *)
        (* Команда 3 *)
        (* … *)

end.
```

Входные переменные
read(x);
readln(x);

Возведение в квадрат
sqr(x) →x^2

Вывод переменных на экран
write(x);→ вывод без перехода на новую строку
writeln(x);→ вывод с переходом на новую строку

«Условный оператор»

Листинг 1. «Нахождение корней квадратного уравнения».

Задание:

Дано квадратное уравнение: $ax^2 + bx + c = 0$. Найти его корни или сообщить, что их нет.

Вводные данные: коэффициенты a, b, c.

Результирующие данные:x (если корень единственный); x1, x2 (если два корня); “Корней в данном уравнении нет”.

Листинг:

```
program Quadratic_equation;

var
  a, b, c, D, x1, x2 : real;

begin

    writeln('ax^2 + bx + c = 0, a <> 0');
writeln('Укажите коэффициенты a, b и c');

write('a = '); readln(a);
    write('b = '); readln(b);
    write('c = '); readln(c);
    D := sqr(b) - 4 * a * c;
    if D < 0 then
    begin
      writeln('Корней в данном уравнении нет')
end
    else
    begin
       if D = 0 then
        begin
          writeln('x = ', - b / (2 * a))
        end
       else
       begin
           x1 := (- b + sqrt(D)) / (2 * a);
           x2 := (- b - sqrt(D)) / (2 * a);
           writeln('x1 = ', x1);
           writeln('x2 = ', x2);
       end;
    end;
end.
```

Тестирование программы:

ax^2 + bx + c = 0, a <> 0 Укажите коэффициенты a, b и c a = 2 b = 5 c = 25 Корней в данном уравнении нет	ax^2 + bx + c = 0, a<> 0 Укажите коэффициенты a, b и c a = 1 b = -2 c = 1 x = 1	ax^2 + bx + c = 0, a<> 0 Укажите коэффициенты a, b и c a = 2 b = 2 c = -12 x1 = 2 x2 = -3

«Условный оператор». Опорный конспект

Общий вид условного оператора

```
if (условие) then
begin
        {Серия команд 1}
end
else
begin
        {Серия команд 2}
end;
```

Неполный условный оператор

```
if (условие) then
begin
        {Команда 1}
        {Команда 2}
        {…}
end
```

Условный оператор без ограничителей begin/end

```
if (условие) then
        {Команда 1}
else
        {Команда 2}
```

Неполный условный оператор без ограничителей begin/end

```
if (условие) then
        {Команда 1}
```

Решение квадратного уравнения

$ax^2 + bx + c = 0, a \neq 0$

$D = b^2 - 4ac$

D < 0, вещественных корней у квадратного уравнения нет.

D > 0, $x = \frac{-b \pm \sqrt{D}}{2a}$

D = 0, $x = \frac{-b}{2a}$

«Составные условия»

Листинг 1. «Месяц (решение с помощью условного оператора if)»

Задание:

Пользователь вводит в программу число – номер месяца. Вывести название месяца, количество дней, время года, сообщить? если месяц не является первым в году, сообщить если месяц не является последним в году.

Листинг:

```
program Number_of_month;
var   month : byte;
begin
write('Введите номер месяца: ');
readln(month);
    if (month < 1) or (month >  12) then
        writeln('Вы ввели неправильный номер месяца!')
else
    begin
        write('Названиемесяца:');
        if month = 1 then writeln('Январь')
        else if month = 2 then writeln('Февраль')
            else if month = 3 then writeln('Март')
                else if month = 4 then writeln('Апрель')
                    else if month = 5 then writeln('Май')
                        else if month = 6 then writeln('Июнь')
                            else if month = 7 then writeln('Июль')
                                else if month = 8 then writeln('Август')
                                    else if month = 9 then writeln('Сентябрь')
                                      else if month = 10 then writeln('Октябрь')
                                          else if month = 11 then writeln('Ноябрь')
                                              elsewriteln('Декабрь');
write('Количество дней в месяце: ');
if (month = 1) or (month = 3) or (month = 5) or (month = 7)or (month = 8) or (month = 10) or (month =
12)
       then
            writeln('31')
        else
if (month = 4) or (month = 6) or (month = 9) or (month = 11) then
                writeln('30')
            else
writeln('28 или 29');
        write('Времягода: ');
        if (month >= 3) and (month <= 5) then
          writeln('Весна')
        else
if (month >= 6) and (month <= 8) then
              writeln('Лето')
            else
if (month >= 9) and (month <= 11) then
                  writeln('Осень')
                else
writeln('Зима');
if not (month = 1) then writeln('Это не первый месяц в году');
        if not (month = 12) then writeln('Это не последний месяц в году');
end;
end.
```

Тестирование программы:

Введите номер месяца: 4 Название месяца:Апрель Количество дней в месяце: 30 Время года: Весна Это не первый месяц в году Это не последний месяц в году	Введите номер месяца: 10 Название месяца: Октябрь Количество дней в месяце: 31 Время года: Осень Это не первый месяц в году Это не последний мсяц в году	Введите номер месяца: 1 Название месяца: Январь Количество дней в месяце: 31 Время года: Зима Это не последний месяц в году

«Составные условия». Опорный конспект

Применение алгебры логики при создании составных условий

Конъюнкция	A ^ B	И	**and**	*Обязательное выполнение всех условий*
Дизъюнкция	A v B	ИЛИ	**or**	*Выполнение хотя бы одного из условий*
Отрицание	$\overline{A}$	НЕ	**not**	*Условие не должно выполняться*

Операции сравнения

Сравнение	Название
=	Равно
<>	Не равно
>	Больше
<	Меньше
>=	Больше или равно
<=	Меньше или равно

Многократно вложенный условный оператор

```
if (условие 1) then
        {Команда 1}
else
        if (условие 2) then
                {Команда 2}
        else
                if (условие 3) then
                        {Команда 3}
                else
                        if(условие 4) then
                                {Команда 4}
                        else
                                …
                                …
                                …                    else {Команда n};
```

«Оператор варианта»

Листинг 1. «Месяц (решение с помощью оператора варианта case)»

Задание:

Пользователь вводит в программу число – номер месяца. Вывести название месяца, количество дней, время года, сообщить? если месяц не является первым в году, сообщить если месяц не является последним в году.
Для вывода названия, количества дней и времени года в алгоритме программы воспользоваться оператором варианта.

Листинг:

```
program Number_of_month;
var
  month : byte;
begin
write('Введите номер месяца: ');
readln(month);
    if (month < 1) or (month >  12) then
        writeln('Вы ввели неправильный номер месяца!')
else
    begin
        write('Названиемесяца:');
        case month of
            1: writeln('Январь');
            2: writeln('Февраль');
            3: writeln('Март');
            4: writeln('Апрель');
            5: writeln('Май');
            6: writeln('Июнь');
            7: writeln('Июль');
            8: writeln('Август');
            9: writeln('Сентябрь');
            10: writeln('Октябрь');
            11: writeln('Ноябрь');
            12: writeln('Декабрь');
end;
write('Количество дней в месяце: ');
case month of
            1, 3, 5, 7, 8, 10, 12: writeln('31');
            4, 6, 9, 11: writeln('30');
            2: writeln('28 или 29');
        end;
        write('Времягода: ');
        case month of
            1, 2,  12: writeln('Зима');
            3, 4, 5: writeln('Весна');
            6, 7, 8: writeln('Лето');
            9, 10, 11: writeln('Осень');
        end;
        if not (month = 1) then writeln('Этонепервыймесяцвгоду');
        if not (month = 12) then writeln('Этонепоследниймесяцвгоду');
end;
end.
```

Тестирование программы:

Введите номер месяца: 9 Название месяца:Сентябрь Количество дней в месяце: 30 Время года: Осень Это не первый месяц в году Это не последний месяц в году	Введите номер месяца: 12 Название месяца:Декабрь Количество дней в месяце: 31 Время года: Зима Это не первый месяц в году	Введите номер месяца: 2 Название месяца: Февраль Количество дней в месяце: 28 или 29 Время года: Зима Это не первый месяц в году Это не последний месяц в году

«Оператор варианта». Опорный конспект

case [проверяемая переменная] **of**

[возможные значения, вариант 1] **:** [операторы если условие 1 выполнено]

[возможные значения, вариант 2] **:** [операторы если условие 2 выполнено]

…

[возможные значения, вариант n] **:** [операторы если условие n выполнено]

else [операторы если ни одно из условий не выполнено]

end;

«Логический тип». Опорный конспект

Название типа – *boolean*.

Диапазон возможных значений – *false / true*.

Составные условия для признаков треугольников

Треугольник - сумма углов равна 180^0. **(ABC+ BAC + ACB = 180)**

Прямоугольный треугольник – один из трёх углов равен 90^0. **(ABC = 90) or (BAC = 90) or (ACB = 90)**

Остроугольный треугольник – каждый из углов меньше 90^0.**(ABC < 90) and (BAC < 90) and (ACB < 90)**

Тупоугольный треугольник – один из углов больше 90^0.**(ABC >90) or (BAC > 90) or (ACB > 90)**

Равносторонний треугольник – каждый из углов равен 60^0. **(ABC = 60) and (BAC = 60) and (ACB = 60)**

Равнобедренный треугольник – хотя бы одна пара угловравны друг другу. **(ABC = BAC) or (BAC = ACB) or (ACB = ABC)**

Разносторонний треугольник – все углы попарно не равны друг другу. **(ABC <> BAC) and (BAC <> ACB) and (ACB <> ABC)**

«Логический тип»

Листинг 1. «Виды треугольников»

Задание:

Для треугольникаABCпользователь вводит углы. Определить вид (или виды) треугольника (прямоугольный, тупоугольный, остроугольный, равносторонний, равнобедренный, разносторонний). Если введённые углы не составляют треугольник, то необходимо вывести соответствующее сообщение.

Вводные данные:угол ABC, угол BAC, угол CAB

Результирующие данные:«прямоугольный», «тупоугольный», «остроугольный», «равносторонний», «равнобедренный», «разносторонний», «Это не углы одного треугольника».

Листинг:

```
program triangle_type;

var
  ABC, BAC, ACB : real;
  triangle : boolean;
  equilateral_triangle : boolean;
  isosceles_triangle : boolean;
  right_triangle : boolean;
  oxygon : boolean;
  obtuse_triangle : boolean;
  scalene_triangle : boolean;

begin

   write('угол ABC: ');readln(ABC);
   write('угол BAC: ');readln(BAC);
   write('угол ACB: '); readln(ACB);

   triangle := (ABC +  BAC + ACB = 180);
   equilateral_triangle := (ABC = 60) and (BAC = 60) and (ACB = 60);
   isosceles_triangle := (ABC = BAC) or (BAC = ACB) or (ACB = ABC);
   right_triangle := (ABC = 90) or (BAC = 90) or (ACB = 90);
   oxygon := (ABC < 90) and (BAC < 90) and (ACB < 90);
   obtuse_triangle := (ABC > 90) or (BAC > 90) or (ACB > 90);
   scalene_triangle := (ABC <> BAC) and (BAC <> ACB) and (ACB <> ABC);

   if not triangle then
      writeln('Этонеуглыодноготреугольника')
   else
   begin
       writeln('характеристикитреугольника:');
(*серия тестов 1 *)
{
       if right_triangle then writeln('прямоугольный');
       if oxygon then writeln('остроугольный');
       if obtuse_triangle then writeln('тупоугольный');
       if equilateral_triangle then writeln('равностороний');
       if isosceles_triangle then writeln('равнобедренный');
```

```
        if scalene_triangle then writeln('разносторонний');
}


(*сериятестов 2 *)
if right_triangle then
          writeln('прямоугольный')
        else if oxygon then
              writeln('остроугольный')
            else if obtuse_triangle then
                  writeln('тупоугольный');

        if equilateral_triangle then
          writeln('равносторонний')
        else if isosceles_triangle then
              writeln('равнобедренный')
            else if scalene_triangle then
                  writeln('разносторонний');

end;
end.
```

Тестирование программы, серия тестов 1:		
угол ABC: 0 угол BAC: 0 угол ACB: 0 Это не углы одного треугольника	угол ABC: 20 угол BAC: 60 угол ACB: 100 характеристики треугольника: тупоугольный разносторонний	угол ABC: 60 угол BAC: 60 угол ACB: 60 характеристики треугольника: остроугольный равносторонний равнобедренный
Тестирование программы, серия тестов 2:		
угол ABC: 90 угол BAC: 30 угол ACB: 60 характеристики треугольника: прямоугольный разносторонний	угол ABC: 20 угол BAC: 20 угол ACB: 140 характеристики треугольника: тупоугольный равнобедренный	угол ABC: 50 угол BAC: 50 угол ACB: 80 характеристики треугольника: остроугольный равнобедренный

«Цикл с параметром»

Листинг 1. «Квадраты натуральных чисел»

Задание:
Вывести квадраты первых 10 целых чисел.
Листинг:

program quadro;
 	
var
  k : integer;
 	
begin
 	
  for k := 1 to 10 do
writeln(k, '^2 = ', sqr(k));
 	
end.

Тестирование программы:
1^2 = 1 2^2 = 4 3^2 = 9 4^2 = 16 5^2 = 25 6^2 = 36 7^2 = 49 8^2 = 64 9^2 = 81 10^2 = 100

Листинг 2. «Квадраты натуральных чисел в обратном порядке»

Задание:
Вывести в обратном порядке квадраты первых 10 целых чисел.
Листинг:

program quadro;
 	
var
  k : integer;
 	
begin
 	
  for k := 10 downto 1 do
writeln(k, '^2 = ', sqr(k));
 	
end.

Тестирование программы:
10^2 = 100 9^2 = 81 8^2 = 64 7^2 = 49 6^2 = 36 5^2 = 25 4^2 = 16 3^2 = 9 2^2 = 4 1^2 = 1

Листинг 3. «Доллар Джона Джонса»

Задание:
В рассказе Гарри Килера «Доллар Джона Джонса» в 1921 году Джон Джонс положил 1 доллар в банк под 3% годовых. Условием вклада было то, что накопившаяся к соответствующему времени сумма достанется 40-му наследнику Джона Джонса. По сюжету рассказа последнее начисление вклада произошло в 2946 году. Чему был равен вклад Джона Джонса за каждый год?
Листинг:

program John_Jones_Dollar;
 	
var
  year : integer;
  sum : real;
 	
const
  rent = 0.03;
  year_first = 1922;
  year_last = 2946;

```
begin

    sum := 1; {Первоначальный депозит Джона Джонса}

for year := year_first to year_last do
    begin
sum := sum *(1 + rent); {Увеличение вклада за текущий год}
writeln('Год: ', year, ' Сумма: ', sum:1:2)
end;

end.
```

Тестирование программы (представлены выборочные результаты):

Год: 1922 Сумма: 1.03 Год: 1923 Сумма: 1.06 Год: 1924 Сумма: 1.09 Год: 1925 Сумма: 1.13 Год: 1926 Сумма: 1.16 Год: 1927 Сумма: 1.19 Год: 1928 Сумма: 1.23 Год: 1929 Сумма: 1.27 Год: 1930 Сумма: 1.30 Год: 1931 Сумма: 1.34 … Год: 2021 Сумма: 19.22 …	
Год: 2121 Сумма: 369.36 …	В рассказе: ≈$19.10
Год: 2221 Сумма: 7098.51 …	В рассказе: ≈$346
Год: 2299 Сумма: 71198.52 …	В рассказе: ≈$6920
Год: 2321 Сумма: 136423.72 …	В рассказе: ≈$68900
Год: 2421 Сумма: 2621877.23 …	В рассказе: ≈$132000
Год: 2521 Сумма: 50388893.66 …	В рассказе: ≈$2520000
Год: 2621 Сумма: 968405603.23 …	В рассказе: ≈$47900000
Год: 2721 Сумма: 18611430896.67 …	В рассказе: ≈$912000000
Год: 2807 Сумма: 236472742848.08 …	В рассказе: ≈$17400000000
Год: 2821 Сумма: 357686241040.31 …	В рассказе: ≈$219000000000
Год: 2921 Сумма: 6874240231169.63 …	В рассказе: ≈$332000000000
Год: 2937 Сумма: 11031137562869.84 Год: 2938 Сумма: 11362071689755.93 Год: 2939 Сумма: 11702933840448.61 Год: 2940 Сумма: 12054021855662.07 Год: 2941 Сумма: 12415642511331.93 Год: 2942 Сумма: 12788111786671.89 Год: 2943 Сумма: 13171755140272.05 Год: 2944 Сумма: 13566907794480.21 Год: 2945 Сумма: 13973915028314.62 Год: 2946 Сумма: 14393132479164.05	В рассказе: ≈$6310000000000

«Цикл с параметром». Опорный конспект

Общий синтаксис

```
for k := a to b do
begin
        {Команда 1;}
        {Команда 2;}
        {Команда 3;}
        …
end;
```

Краткая форма (если внутри тела цикла только одна команда)

```
for k := a to b do
        {Команда 1;}
```

«Вложенные циклы». Опорный конспект

Перебор всех возможных комбинаций (x, y, z)

```
for x := x1 to x2 do
        for y := y1 to y2 do
                for z := z1 to z2 do
                begin
                        {Тело самого внутреннего цикла}
                end;
```

«Вложенные циклы»

Листинг 1. «Пифагоровы тройки»

Задание:

Вывести все тройки целых чисел, для которых верно равенство: $\mathbf{a}^2 + \mathbf{b}^2 = \mathbf{c}^2$.
Числа a, b и c брать в пределах от 1 до 100.

Листинг:

Первоначальный вариант

```
program Pythagorean_triples;

var
   a, b, c, S, v : integer;

begin

S := 0; {Пока что найдено 0 троек}
v := 0; {Пока что проверено 0 вариантов}

for c := 1 to 100 do
        for b := 1 to 100 do
            for a := 1 to 100 do
            begin
                inc(v);
                if sqr(a) + sqr(b) = sqr(c) then
                begin
                    writeln(a, ' ', b, ' ', c);
                    s := s + 1;
                end;
            end;

    writeln('Нашли ', S, ' пифагоровыхтроек');
writeln('При этом рассмотрено', v, ' вариантов');

end.
```

Оптимизированный вариант

```
program Pythagorean_triples;

var
   a, b, c, S, v : integer;

begin

S := 0; {Пока что найдено 0 троек}
v := 0; {Пока что проверено 0 вариантов}

for c := 3 to 100 do
        for b := 2 to c - 1 do
        for a := 1 to b - 1 do
            begin
                inc(v);
                if sqr(a) + sqr(b) = sqr(c) then
                begin
                    writeln(a, ' ', b, ' ', c);
                    s := s + 1;
                end;
            end;

    writeln('Нашли ', S, ' пифагоровыхтроек');
    writeln('При этом рассмотрено', v, '
вариантов');

end.
```

Тестирование программы (выборочные результаты):

```
4 3 5
3 4 5
8 6 10
6 8 10
12 5 13
5 12 13
…
84 35 91
35 84 91
76 57 95
57 76 95
72 65 97
65 72 97
96 28 100
80 60 100
60 80 100
28 96 100
Нашли104пифагоровыхтроек
При этом рассмотрено 1000000 вариантов
```

```
3 4 5
6 8 10
5 12 13
9 12 15
8 15 17
12 16 20
…
36 77 85
13 84 85
60 63 87
39 80 89
54 72 90
35 84 91
57 76 95
65 72 97
60 80 100
28 96 100
Нашли52пифагоровыхтроек
При этом рассмотрено161700вариантов
```

«Цикл с предусловием»

Листинг 1. «Алгоритм Евклида»

Задание:

Методом Евклида рассчитать и вывести наибольший общий делитель для двух целых положительных чисел.

Вводные данные:два целых положительных числа, аи b.

Результирующие данные: наибольший общий делитель.

Листинг:

```
program Euclids_algorithm;

var
  a, b : integer;

begin

   write('Введитепервоечисло a: '); readln(a);
   write('Введите второе число b: '); readln(b);

while (a <> 0) and (b <> 0) do
   begin
      if a > b then
      begin
         write('a = остаток(', a, '/', b, ') = ');
         a := a mod b;
         writeln(a, ', b = ', b);
      end
      else
      begin
         write('b = остаток(', b, '/', a, ') = ');
         b := b mod a;
         writeln(b, ', a = ', a);
      end;
end;

writeln('Наибольший общий делитель: ', a + b);

end.
```

Тестирование программы:

Введите первое число a: 120 Введите второе число b: 72 a = остаток(120/72) = 48, b = 72 b = остаток(72/48) = 24, a = 48 a = остаток(48/24) = 0, b = 24 Наибольший общий делитель: 24	Введите первое число a: 42 Введите второе число b: 144 b = остаток(144/42) = 18, a = 42 a = остаток(42/18) = 6, b = 18 b = остаток(18/6) = 0, a = 6 Наибольший общий делитель: 6

«Цикл с предусловием». Опорный конспект

Основной синтаксис

```
while (условие выполнения цикла) do
begin
    {Команда 1;}
    {Команда 2;}
    …
    {Команда n;}
end;
```

Краткая форма (если в теле цикла только одна команда)

```
while (условие выполнения цикла) do
    {Команда 1;}
```

Особенности

1. Цикл может не выполниться ни разу.
2. При неправильно составленном алгоритме цикл может стать бесконечным.

«Цикл с постусловием». Опорный конспект

Основной синтаксис

```
repeat
        {Команда 1;}
        {Команда 2;}
        …
        {Команда n;}
until (Условие выхода из цикла);
```

Особенности

1. Цикл выполнится минимум 1 раз.
2. При неправильно составленном алгоритме цикл может стать бесконечным.
3. Служебные слова begin/end для ограничения тела цикла не используются.

Выбор цикла

```
if (Известно количество повторений) then
        {Используйте цикл FOR}
else
    if(Известно что цикл будет выполняться как минимум один раз)then
            {Используйте цикл REPEAT}
    else
            {Используйте цикл WHILE}
```

«Цикл с постусловием»

Листинг 1. «Вычисление числа π с помощью ряда Лейбница»

Задание:

Вычислить число πдо одной десятимиллионной с помощью ряда Лейбница.Подсчитать, при этом, сколько потребуется вычислить членов последовательности для достижения необходимой точности.

$$\pi = \frac{4}{1} - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \frac{4}{9} - \cdots$$

Листинг:

```
program Life_of_Pi;

const
    accuracy = 0.0000001;

var
  pi : real;
  n : integer;
  member : real;

begin

    pi := 0;
    n := 1;

    repeat

        member := 4 / (2 * n - 1);

        if n mod 2 = 1 then
          pi := pi + member
        else
           pi := pi - member;

        inc(n);

    until (member < accuracy);

    writeln('Число Пи: ', pi);
writeln('Количество итераций цикла: ', n);

end.
```

Тестирование программы:

Число Пи: 3.14159270358986
Количество итераций цикла: 20000002

«Символьный тип»

Листинг 1. «Все символы символьного типа и их коды»

Задание:

Вывести все литеры типа char. Возле каждого символа указать его код.

Листинг:

```
programchars_examples;
var
ch : char;
k: byte;
begin
fork := 0 to 255 do
begin
ch := chr(k);
writeln('Символ: ', ch, ' Кодсимвола: ', k);
end;
end.
```

Тестирование программы:

Символ:Код символа: 0
Символ: Код символа: 1
Символ:Код символа: 2
Символ:□ Код символа: 3
Символ:□Код символа: 4
Символ:□ Код символа: 5
Символ:□ Код символа: 6
Символ:□ Код символа: 7
Символ:□ Код символа: 8
Символ:
Код символа: 9
Символ:
 Код символа: 10
Символ: □ Код символа: 11
Символ: □ Код символа: 12
Символ:
 Код символа: 13
Символ: □ Код символа: 14
Символ:□ Код символа: 15
Символ:□ Код символа: 16
Символ:□ Код символа: 17
Символ: □ Код символа: 18
Символ:□ Код символа: 19
Символ:□ Код символа: 20
Символ: □ Код символа: 21
Символ: 5 Код символа: 53
Символ: 6 Код символа: 54
Символ: 7 Код символа: 55
Символ: 8 Код символа: 56
Символ: 9 Код символа: 57
Символ: : Код символа: 58
Символ: ; Код символа: 59
Символ: < Код символа: 60
Символ: = Код символа: 61
Символ: > Код символа: 62
Символ: ? Код символа: 63
Символ: @ Код символа: 64
Символ: A Код символа: 65
Символ: B Код символа: 66
Символ: C Код символа: 67
Символ: D Код символа: 68
Символ: E Код символа: 69
Символ: F Код символа: 70
Символ: G Код символа: 71
Символ: H Код символа: 72
Символ: I Код символа: 73
Символ: J Код символа: 74
Символ: m Код символа: 109
Символ: n Код символа: 110
Символ: o Код символа: 111
Символ: p Код символа: 112
Символ: q Код символа: 113
Символ: r Код символа: 114
Символ: s Код символа: 115
Символ: t Код символа: 116
Символ: u Код символа: 117
Символ: v Код символа: 118
Символ: w Код символа: 119
Символ: x Код символа: 120
Символ: y Код символа: 121
Символ: z Код символа: 122
Символ: { Код символа: 123
Символ: | Код символа: 124
Символ: } Код символа: 125
Символ: ~ Код символа: 126
Символ: □ Код символа: 127
Символ: Ђ Код символа: 128
Символ: Ѓ Код символа: 129
Символ: ‚ Код символа: 130
Символ: Ґ Код символа: 165
Символ: ¦ Код символа: 166
Символ: § Код символа: 167
Символ: Ё Код символа: 168
Символ: © Код символа: 169
Символ: Є Код символа: 170
Символ: « Код символа: 171
Символ: ¬ Код символа: 172
Символ: - Код символа: 173
Символ: ® Код символа: 174
Символ: Ї Код символа: 175
Символ: ° Код символа: 176
Символ: ± Код символа: 177
Символ: І Код символа: 178
Символ: і Код символа: 179
Символ: ґ Код символа: 180
Символ: µ Код символа: 181
Символ: ¶ Код символа: 182
Символ: · Код символа: 183
Символ: ё Код символа: 184
Символ: № Код символа: 185
Символ: є Код символа: 186
Символ: Э Код символа: 221
Символ: Ю Код символа: 222
Символ: Я Код символа: 223
Символ: а Код символа: 224
Символ: б Код символа: 225
Символ: в Код символа: 226
Символ: г Код символа: 227
Символ: д Код символа: 228
Символ: е Код символа: 229
Символ: ж Код символа: 230
Символ: з Код символа: 231
Символ: и Код символа: 232
Символ: й Код символа: 233
Символ: к Код символа: 234
Символ: л Код символа: 235
Символ: м Код символа: 236
Символ: н Код символа: 237
Символ: о Код символа: 238

Символ: □Код символа: 22
Символ: □Код символа: 23
Символ: □Код символа: 24
Символ: □Код символа: 25
Символ: □Код символа: 26
Символ:□ Код символа: 27
Символ: □Код символа: 28
Символ: □Код символа: 29
Символ: - Код символа: 30
Символ: □ Код символа: 31
Символ: □ Код символа: 32
Символ: ! Код символа: 33
Символ: " Код символа: 34
Символ: # Код символа: 35
Символ: $ Код символа: 36
Символ: % Код символа: 37
Символ: & Код символа: 38
Символ: ' Код символа: 39
Символ: (Код символа: 40
Символ:) Код символа: 41
Символ: * Код символа: 42
Символ: + Код символа: 43
Символ: , Код символа: 44
Символ: - Код символа: 45
Символ: . Код символа: 46
Символ: / Код символа: 47
Символ: 0 Код символа: 48
Символ: 1 Код символа: 49
Символ: 2 Код символа: 50
Символ: 3 Код символа: 51
Символ: 4 Код символа: 52

Символ: K Код символа: 75
Символ: L Код символа: 76
Символ: M Код символа: 77
Символ: N Код символа: 78
Символ: O Код символа: 79
Символ: P Код символа: 80
Символ: Q Код символа: 81
Символ: R Код символа: 82
Символ: S Код символа: 83
Символ: T Код символа: 84
Символ: U Код символа: 85
Символ: V Код символа: 86
Символ: W Код символа: 87
Символ: X Код символа: 88
Символ: Y Код символа: 89
Символ: Z Код символа: 90
Символ: [Код символа: 91
Символ: \ Код символа: 92
Символ:] Код символа: 93
Символ: ^ Код символа: 94
Символ: _ Код символа: 95
Символ: ` Код символа: 96
Символ: a Код символа: 97
Символ: b Код символа: 98
Символ: c Код символа: 99
Символ: d Код символа: 100
Символ: e Код символа: 101
Символ: f Код символа: 102
Символ: g Код символа: 103
Символ: h Код символа: 104
Символ: i Код символа: 105
Символ: j Код символа: 106
Символ: k Код символа: 107
Символ: l Код символа: 108

Символ: ѓ Код символа: 131
Символ: „ Код символа: 132
Символ: … Код символа: 133
Символ: † Код символа: 134
Символ: ‡ Код символа: 135
Символ: € Код символа: 136
Символ: ‰ Код символа: 137
Символ: Љ Код символа: 138
Символ: ‹ Код символа: 139
Символ: Њ Код символа: 140
Символ: Ќ Код символа: 141
Символ: Ћ Код символа: 142
Символ: Џ Код символа: 143
Символ: ђ Код символа: 144
Символ: ‘ Код символа: 145
Символ: ’ Код символа: 146
Символ: “ Код символа: 147
Символ: ” Код символа: 148
Символ: • Код символа: 149
Символ: – Код символа: 150
Символ: — Код символа: 151
Символ: □ Код символа: 152
Символ: ™ Код символа: 153
Символ: љ Код символа: 154
Символ: › Код символа: 155
Символ: њ Код символа: 156
Символ: ќ Код символа: 157
Символ: ћ Код символа: 158
Символ: џ Код символа: 159
Символ: Код символа: 160
Символ: Ў Код символа: 161
Символ: ў Код символа: 162
Символ: J Код символа: 163
Символ: ¤ Код символа: 164

Символ: » Код символа: 187
Символ: j Код символа: 188
Символ: S Код символа: 189
Символ: s Код символа: 190
Символ: ї Код символа: 191
Символ: А Код символа: 192
Символ: Б Код символа: 193
Символ: В Код символа: 194
Символ: Г Код символа: 195
Символ: Д Код символа: 196
Символ: Е Код символа: 197
Символ: Ж Код символа: 198
Символ: З Код символа: 199
Символ: И Код символа: 200
Символ: Й Код символа: 201
Символ: К Код символа: 202
Символ: Л Код символа: 203
Символ: М Код символа: 204
Символ: Н Код символа: 205
Символ: О Код символа: 206
Символ: П Код символа: 207
Символ: Р Код символа: 208
Символ: С Код символа: 209
Символ: Т Код символа: 210
Символ: У Код символа: 211
Символ: Ф Код символа: 212
Символ: Х Код символа: 213
Символ: Ц Код символа: 214
Символ: Ч Код символа: 215
Символ: Ш Код символа: 216
Символ: Щ Код символа: 217
Символ: Ъ Код символа: 218
Символ: Ы Код символа: 219
Символ: Ь Код символа: 220

Символ: п Код символа: 239
Символ: р Код символа: 240
Символ: с Код символа: 241
Символ: т Код символа: 242
Символ: у Код символа: 243
Символ: ф Код символа: 244
Символ: х Код символа: 245
Символ: ц Код символа: 246
Символ: ч Код символа: 247
Символ: ш Код символа: 248
Символ: щ Код символа: 249
Символ: ъ Код символа: 250
Символ: ы Код символа: 251
Символ: ь Код символа: 252
Символ: э Код символа: 253
Символ: ю Код символа: 254
Символ: я Код символа: 255

Листинг 2. «Большие буквы латинского алфавита»

Задание:

Вывести все символы являющиеся заглавными буквами латинского алфавита. Возле каждого символа написать его код.

Листинг:

```
program chars_examples;

var
  ch : char;
  k: byte;

begin

   for ch := 'A' to 'Z' do
   begin
     k := ord(ch);
     writeln('Символ: ', ch, ' Кодсимвола: ', k);
end;

end.
```

Тестирование программы:

```
Символ: A Код символа: 65
Символ: B Код символа: 66
Символ: C Код символа: 67
Символ: D Код символа: 68
Символ: E Код символа: 69
Символ: F Код символа: 70
Символ: G Код символа: 71
Символ: H Код символа: 72
Символ: I Код символа: 73
Символ: J Код символа: 74
Символ: K Код символа: 75
Символ: L Код символа: 76
Символ: M Код символа: 77
Символ: N Код символа: 78
Символ: O Код символа: 79
Символ: P Код символа: 80
Символ: Q Код символа: 81
Символ: R Код символа: 82
Символ: S Код символа: 83
Символ: T Код символа: 84
Символ: U Код символа: 85
Символ: V Код символа: 86
Символ: W Код символа: 87
Символ: X Код символа: 88
Символ: Y Код символа: 89
Символ: Z Код символа: 90
```

Листинг 3. «Малые буквы латинского алфавита»

Задание:

Вывести все символы являющиеся строчными буквами латинского алфавита. Возле каждого символа написать его код.

Листинг:

```
program chars_examples;

var
  ch : char;
  k: byte;

begin

   for ch := 'a' to 'z' do
   begin
     k := ord(ch);
     writeln('Символ: ', ch, ' Кодсимвола: ',
k);
end;

end.
```

Тестирование программы:

```
Символ: a Код символа: 97
Символ: b Код символа: 98
Символ: c Код символа: 99
Символ: d Код символа: 100
Символ: e Код символа: 101
Символ: f Код символа: 102
Символ: g Код символа: 103
Символ: h Код символа: 104
Символ: i Код символа: 105
Символ: j Код символа: 106
Символ: k Код символа: 107
Символ: l Код символа: 108
Символ: m Код символа: 109
Символ: n Код символа: 110
Символ: o Код символа: 111
Символ: p Код символа: 112
Символ: q Код символа: 113
Символ: r Код символа: 114
Символ: s Код символа: 115
Символ: t Код символа: 116
Символ: u Код символа: 117
Символ: v Код символа: 118
Символ: w Код символа: 119
Символ: x Код символа: 120
Символ: y Код символа: 121
Символ: z Код символа: 122
```

Листинг 4. «Большие буквы русского алфавита»

Задание:

Вывести все символы являющиеся заглавными буквами русского алфавита. Возле каждого символа написать его код.

Листинг:

```
program chars_examples;

var
  ch : char;
  k: byte;

begin

   for ch := 'А' to 'Я' do
   begin
     k := ord(ch);
     writeln('Символ: ', ch, ' Кодсимвола: ', k);
end;

end.
```

Тестирование программы:

```
Символ: А Код символа: 192
Символ: Б Код символа: 193
Символ: В Код символа: 194
Символ: Г Код символа: 195
Символ: Д Код символа: 196
Символ: Е Код символа: 197
Символ: Ж Код символа: 198
Символ: З Код символа: 199
Символ: И Код символа: 200
Символ: Й Код символа: 201
Символ: К Код символа: 202
Символ: Л Код символа: 203
Символ: М Код символа: 204
Символ: Н Код символа: 205
Символ: О Код символа: 206
Символ: П Код символа: 207
Символ: Р Код символа: 208
Символ: С Код символа: 209
Символ: Т Код символа: 210
Символ: У Код символа: 211
Символ: Ф Код символа: 212
Символ: Х Код символа: 213
Символ: Ц Код символа: 214
Символ: Ч Код символа: 215
Символ: Ш Код символа: 216
Символ: Щ Код символа: 217
Символ: Ъ Код символа: 218
Символ: Ы Код символа: 219
Символ: Ь Код символа: 220
Символ: Э Код символа: 221
Символ: Ю Код символа: 222
Символ: Я Код символа: 223
```

Листинг 5. «Малые буквы русского алфавита»

Задание:

Вывести все символы являющиеся строчными буквами русского алфавита. Возле каждого символа написать его код.

Листинг:

```
program chars_examples;

var
  ch : char;
  k: byte;

begin

   for ch := 'а' to 'я' do
   begin
     k := ord(ch);
     writeln('Символ: ', ch, ' Кодсимвола: ',
k);
end;

end.
```

Тестирование программы:

```
Символ: а Код символа: 224
Символ: б Код символа: 225
Символ: в Код символа: 226
Символ: г Код символа: 227
Символ: д Код символа: 228
Символ: е Код символа: 229
Символ: ж Код символа: 230
Символ: з Код символа: 231
Символ: и Код символа: 232
Символ: й Код символа: 233
Символ: к Код символа: 234
Символ: л Код символа: 235
Символ: м Код символа: 236
Символ: н Код символа: 237
Символ: о Код символа: 238
Символ: п Код символа: 239
Символ: р Код символа: 240
Символ: с Код символа: 241
Символ: т Код символа: 242
Символ: у Код символа: 243
Символ: ф Код символа: 244
Символ: х Код символа: 245
Символ: ц Код символа: 246
Символ: ч Код символа: 247
Символ: ш Код символа: 248
Символ: щ Код символа: 249
Символ: ъ Код символа: 250
Символ: ы Код символа: 251
Символ: ь Код символа: 252
Символ: э Код символа: 253
Символ: ю Код символа: 254
Символ: я Код символа: 255
```

«Символьный тип». Опорный конспект

Название типа→char

Количество элементов в типе→ 256

Символы бывают **печатные** (буквы, спецсимволы, цифры и проч.) и **управляющие** (возврат каретки, переход на новую строку, пустой символ и т.д.)..

Основные функции:

ord(ch) → для символа ch возвращается его символьный код (число от 0 до 255)

chr(n) → для числаn возвращается соответствующей ему символ

pred(ch) → возвращает символ у которого символьный код меньше на единицу

succ(ch) → возвращаетсимвол у которого символьный код больше на единицу

«Строки», «Обработка строк». Опорный конспект

Название типа → string

Любая строка представляет из себя упорядоченный набор символов.

Основные функции

str[k] →возвращает k-й элемент-символ строки str

pos(str1, str2) →возвращает номер позициив строке str2, с которой начинается строка str1

copy(str, k, m) →возвращает из строки str подстроку,длиной mи начинающейся в строке strc позиции k

delete(str, k, m) →удаляет из строки str подстроку,длиной mи начинающейся в строке strc позиции k

insert(str1, str2, k) →вставляет, начиная с позиции k, строку str1 в строкуstr2

concat(str1, str2, str3, …) →объединяет строки str1, str2, str3, … в одну строку

«Строки»

Листинг 1. «Строковые функции и процедуры»

Задание:

Продемонстрировать различные функции и процедуры для работы со строковым типом данных.

Листинг:

```
program string_examples;

var
  k: integer;
  str, str1, str2, s : string;
  comparison : boolean;

begin
    (* Первыйтест *)
    str1 := 'Большая строка';
str2 := 'А это (с точки зрения языка Паскаль) меньшая строка!';
comparison := str1 > str2;writeln(comparison);

str := 'Строка в языке Паскаль - это не что иное как одномерный массив символов';

(* Второйтест *)
for k := 1 to length(str) do
    begin
        write(str[k]);
         if k <> length(str) then write(', ') else writeln;
    end;

(* Третийтест *)
    k := pos('Паскаль', str);
    writeln(k);
(* Четвёртыйтест *)
    k := pos('Pascal', str);
    writeln(k);

    (* Пятыйтест *)
    s := copy(str, 16, 7);
    writeln(s);
    (* Шестойтест *)
    s := str[1];
    writeln(s);

    (* Седьмойтест *)
    s := copy(str, 16, 20);
    writeln(s);

    (* Восьмойтест *)
    delete(s, 17, 1);
    writeln(s);

    (* Девятыйтест *)
    insert('всегда ', s, 15);
    writeln(s);

    (* Десятыйтест *)
```

```
    s := concat('Язык ', s, '!');
    writeln(s);
  end.
```

Тестирование программы:

1-й тест:

True

2-й тест:

С, т, р, о, к, а, , в, , я, з, ы, к, е, , П, а, с, к, а, л, ь, , -, , э, т, о, , н, е, , ч, т, о, , и, н, о, е, , к, а, к, , о, д, н, о, м, е, р, н, ы, й, , м, а, с, с, и, в, , с, и, м, в, о, л, о, в

3-й тест:

16

5-й тест:

Паскаль

4-й тест:

0

6-й тест:

С

7-й тест:

Паскаль – это не что

8-й тест:

Паскаль – это нечто

9-й тест:

Паскаль – это всегда нечто

10-й тест:

Паскаль – это всегда нечто!

«Обработка строк»

Листинг 1. «Палиндром»

Задание:

Проверить, является ли строка *строгим палиндромом*. То есть совпадает ли первый символ с последним, второй с предпоследним и т.д.

Вводные данные:строка s.

Результирующие данные:«Это палиндром!»/ «Это не палиндром!»

Листинг:

```
programPalindrome;

vars : string;
is_palindrome : boolean;
k : byte;

begin

   write('Введитетекст: ');readln(s);
   is_palindrome := true;

   for k := 1 to length(s) div 2 do
      if s[k] <> s[length(s) - k + 1] then begin
         is_palindrome := false;
         writeln('Этонепалиндром!');
         break;
      end;
   if is_palindrome then writeln('Этопалиндром!');

end.
```

Тестирование программы:

Во храпе толп – оплот епархов.
Это не палиндром!

лезу в узел
Это палиндром!

мат и тут и там
Это палиндром!

«Функции»

Листинг 1. «Палиндром»

Задание:

Проверить, является ли строка *нестрогим палиндромом*. То есть совпадает ли первый символ с последним, второй с предпоследним и т.д. При проверке не должны иметь значение пробелы и знаки препинания. Регистры букв не обрабатываются. Для этого в тестах использовать фразы, состоящие только из заглавных или только из строчных букв.

Вводные данные:строка s.

Результирующие данные:«Это палиндром!»/ «Это не палиндром!»

Листинг:

```
programPalindrome;

vars : string;
is_palindrome : boolean;
k : byte;
function Delete_character(s: string; l: char): string;
var  m: byte;
begin
   m := 1;
   repeat
       if copy(s, m, 1) = l then
          delete(s, m, 1)
       else
          inc(m);
   until m > length(s);
   Delete_character := s;
end;
begin
   write('Введитетекст: ');readln(s);
   is_palindrome := true;
   s := Delete_character(s, ' '); s := Delete_character(s, '.'); s := Delete_character(s, ',');
   s := Delete_character(s, '-'); s := Delete_character(s, '!'); s := Delete_character(s, '?');
   s := Delete_character(s, ':'); s := Delete_character(s, ';'); s := Delete_character(s, '"');

   for k := 1 to length(s) div 2 do
      if s[k] <> s[length(s) - k + 1] then begin
         is_palindrome := false;
         writeln('Этонепалиндром!');
         break;
      end;
   if is_palindrome then writeln('Этопалиндром!');
end.
```

Тестирование программы:

```
не до логики – голоден
Это палиндром!
```

```
муза! ранясь шилом опыта, ты помолишься на разум
Это палиндром!
```

```
нече выть, ты - вечен
Это палиндром!
```

«Функции». Опорный конспект

functionname_function ([входные данные]) **:** [тип возвращаемого значения];
{Раздел описаний функции}
begin
{Операторы}
name_function**:=** [Возвращаемое значение];
end;

Функция всегда возвращает значение, которые вычисляется на основании входных данных.

«Процедуры». Опорный конспект

procedurename_procedure ([входныеданные]);
{Раздел описаний процедуры}
begin
{Операторы}
end;

В отличие от функций, процедуры не возвращают конкретной значение, но, на основании входных данных выполняют некоторые действия

«Рекурсивные функции», «Рекурсивные процедуры». Опорный конспект

Рекурсивные функции и процедуры для достижения результата вызывают (как правило, многократно) сами себя.

Любой циклический алгоритм можно преобразовать в рекурсивный.

Как и циклы, неправильно составленная рекурсия может оказаться «бесконечным» процессом. Поэтому нужно предусматривать условие завершения рекурсии.

Обычно рекурсия позволяет составлять более простые алгоритмы чем циклы. С другой стороны, рекурсивные алгоритмы потребляют больше ресурсов компьютера, по сравнению с циклическими.

.

«Процедуры»

Листинг 1. «Палиндром»

Задание:

Проверить, является ли строка *палиндромом*. То есть совпадает ли первый символ с последним, второй с предпоследним и т.д. При проверке не должны иметь значение пробелы и знаки препинания. Регистры обрабатываются, то есть заглавные и строчные буквы являются эквивалентными друг другу.

Вводные данные:строка s.

Результирующие данные:«Это палиндром!»/ «Это не палиндром!»

Листинг:

```
programPalindrome;

vars : string;
is_palindrome : boolean;
k : byte;

procedure string_to_lower (first_ord, last_ord, step: integer);
var
  new_chr: char;
  k: byte;
begin
    for k := 1 to length(s) do
    begin
        if (ord(s[k]) >= first_ord) and (ord(s[k]) <= last_ord) then
        begin
            new_chr := chr(ord(s[k]) + step);
            delete(s, k, 1);
            insert(new_chr, s, k);
        end;
    end;
end;

function Delete_character(s: string; l: char): string;
var
m: byte;
begin
   m := 1;
   repeat
        if copy(s, m, 1) = l then
            delete(s, m, 1)
        else
           inc(m);
   until m > length(s);
   Delete_character := s;
end;

begin

   write('Введитетекст: ');
```

```
  readln(s);
    is_palindrome := true;



  s := Delete_character(s, ' ');
  s := Delete_character(s, '.');
  s := Delete_character(s, ',');
  s := Delete_character(s, '-');
  s := Delete_character(s, '!');
  s := Delete_character(s, '?');
    s := Delete_character(s, ':');
   s := Delete_character(s, ';');
   s := Delete_character(s, '"');

    string_to_lower(192, 223, 32);

    for k := 1 to length(s) div 2 do
      if s[k] <> s[length(s) - k + 1] then begin
        is_palindrome := false;
        writeln('Этонепалиндром!');
        break;
      end;
    if is_palindrome then writeln('Этопалиндром!');

  end.
```

Тестирование программы:

Коли мили в шагу, жди Джугашвили, милок.
Это палиндром!

Дорого небо, да надобен огород.
Это палиндром!

Ниша метро…. Автор – крот в аорте машин.
Это палиндром!

«Рекурсивные функции»

Листинг 1. «Числа Фибоначчи»

Задание:

Подсчитать и вывести на экран первые nчисел Фибоначчи. Числа Фибоначчи вычисляются с помощью рекуррентного соотношения F(i) = F(i-1) + F(i-2), F(1) = 1, F(2) = 1.

Вводные данные:n

Результирующие данные:

1-е число Фибоначчи: 1
2-е число Фибоначчи: 1
3-е число Фибоначчи: 2
4-е число Фибоначчи: 3
5-е число Фибоначчи: 5
…
n-е число Фибоначчи: …

Листинг:

```
programget_fibonacci;

var
  n , k: byte;

function fibonacci(m: byte): integer;
begin
   if (m = 1) or (m = 2) then
      fibonacci := 1
   else
      fibonacci := fibonacci(m - 1) + fibonacci(m - 2);
end;

begin

write('СколькочиселФибоначчинайти?'); readln(n);
   for k := 1 to n do   writeln(k, '-ечислоФибоначчи: ', fibonacci(k));

end.
```

Тестирование программы:

Сколько чисел Фибоначчи найти? 15
1-е число Фибоначчи: 1
2-е число Фибоначчи: 1
3-е число Фибоначчи: 2
4-е число Фибоначчи: 3
5-е число Фибоначчи: 5
6-е число Фибоначчи: 8
7-е число Фибоначчи: 13
8-е число Фибоначчи: 21
9-е число Фибоначчи: 34
10-е число Фибоначчи: 55
11-е число Фибоначчи: 89
12-е число Фибоначчи: 144
13-е число Фибоначчи: 233
14-е число Фибоначчи: 377
15-е число Фибоначчи: 610

«Рекурсивные процедуры»

Листинг 1. «Ханойская башня (рекурсивный способ)»

Задание:

Запрограммировать с помощью рекурсии решение головоломки «Ханойская башня». Условие задачи: имеются 3 стержня, на один из которых нанизаны диски разного диаметра. Диски более большего размера находятся под дисками меньшего размера. Диски можно перемещать с одного стержня на другой. При этом за один раз можно переносить только один диск и не разрешается диски больше размера класть на диски меньшего размера. Задание: перенести все диски с 1-го стержня на 3-й.

Вводные данные:количество дисков n.

Результирующие данные:порядок перемещения дисков между стержнями 1, 2 и 3.

Листинг:

```
program Tower_of_Hanoi;

var n : byte;

procedure Hanoi(first, second, third, m : byte);
begin
    if m = 1 then
      writeln(first,' --> ',third,'; ')
    else
    begin
        Hanoi(first,third,second,m - 1);
        Hanoi(first, second, third, 1);
        Hanoi(second, first, third, m-1);
    end;
end;

begin

    write('n = ');
readln(n);
    Hanoi(1, 2, 3, n);

end.
```

Тестирование программы:

```
n = 3
1 --> 3;
1 --> 2;
3 --> 2;
1 --> 3;
2 --> 1;
2 --> 3;
1 --> 3;
```

```
n = 4
1 --> 2;
1 --> 3;
2 --> 3;
1 --> 2;
3 --> 1;
3 --> 2;
1 --> 2;
1 --> 3;
2 --> 3;
2 --> 1;
3 --> 1;
2 --> 3;
1 --> 2;
1 --> 3;
2 --> 3;
```

Листинг 2. «Ханойская башня (циклический способ)»

Задание:

Запрограммировать с помощью циклов решение головоломки «Ханойская башня».

Вводные данные:количество дисков n.

Результирующие данные:порядок перемещения дисков между стержнями 1, 2 и 3.

Листинг:

```
program Hanoi_of_Tower;
var n : integer;
procedure Hanoi(n : longint);
var
   m, k, l, s : integer;
   iz, v, c, x : byte;
begin
    n := longint(round(exp(n * ln(2)))) - 1;
    for m := 1 to n do
    begin
        iz := 1; c := 2; v := 3; k := 1; l := n; s := (k + l) div 2;
        repeat
            if m < s then
              begin
                  x := v;  v := c;  c := x; l := s - 1;
end;
            if m > s then
              begin
                  x := iz; iz := c; c := x; k := s + 1;
end;
            if m = s then
              begin
                 writeln(iz,' ---> ',v); break;
end;
            s := (k + l) div 2;
        until false;
    end;
end;

begin
    write('n = ');
readln(n);
    Hanoi(n);
end.
```

Тестирование программы:

```
n = 3
1 --> 3;
1 --> 2;
3 --> 2;
1 --> 3;
2 --> 1;
2 --> 3;
1 --> 3;
```

«Одномерные массивы». Опорный конспект

var

{имя переменной-массива} **: array** [диапазон индексов] **of** {тип}

25	98	41	14	78
1	2	3	4	5

name_of_array[2] → 98

«Двухмерные массивы». Опорный конспект

var

{имя переменной-массива} **: array** [диапазон индексов 1, диапазон индексов 2] **of** {тип}

11	10	99
25	33	24
18	31	48

name_of_array[1, 1] → 11

name_of_array[1, 2] → 10

name_of_array[2, 1] → 25

name_of_array[2, 3] → 24

Многомерный массив

var

{имяпеременной-массива} **: array** [диапазониндексов 1, диапазониндексов 2, …, диапазониндексовn] **of** {тип}

name_of_array[i_1,i_2, …, i_n] →обращение к элементу с координатами (i_1, i_2, …, i_n) вn-мерном массиве

«Одномерные массивы»

Листинг 1. «Обработка одномерных массивов»

Задание:

Продемонстрировать примеры обработки одномерных массивов.

1) Создать процедуру, генерирующую массив.
2) Создать процедуру, печатающую по порядку элементы массива.
3) Создать процедуру, подсчитывающую сумму элементов массива.
4) Создать процедуру, которая находит минимальный и максимальный элементы массива и меняет их местами.

Листинг:

```
program Working_with_arrays;

const
n = 20;

var
  arr : array[1..n] of integer;
  k : byte;

procedure array_fill;
begin

    Randomize;
    for k := 1 to n do
arr[k] := 10 + random(90);

end;


procedure array_print;
begin

    for k := 1 to n do
write(arr[k], ' ');
    writeln;

end;


procedure search_sum;

var
sum : integer;

begin

    sum := 0;
    for k := 1 to n do
sum := sum + arr[k];
    writeln('Сумма элементов массива: ', sum);
```

```
end;

procedure search_min_max;

var
  min, max: integer;
mn, mx : byte;

begin

    mn := 1;
min := arr[mn];

    mx := 1;
max := arr[mx];

    for k := 2 to n do
    begin
       if arr[k] < min then
       begin
           min := arr[k];
 mn := k;
       end;
       if arr[k] > max then
       begin
           max := arr[k];
mx := k;
       end;
    end;

    writeln('Min: arr[', mn, '] = ', min, ' Max: arr[', mx, '] = ', max);
    arr[mn] := max;  arr[mx] := min;
array_print;

end;

 begin
    array_fill;
    array_print;
    {Первыйтест}search_sum;
{Второйтест}search_min_max;
end.
```

Тестирование программы:

1-й тест:

94 80 23 22 23 14 60 53 64 84 29 52 94 92 87 76 85 96 30 95
Сумма элементов массива: 1253

2-й тест:

17 12 96 **11** 19 12 **99** 91 24 51 26 56 38 28 17 63 58 17 53 67
Min: arr[4] = 11 Max: arr[7] = 99
17 12 96 **99** 19 12 **11** 91 24 51 26 56 38 28 17 63 58 17 53 67

«Сортировка массивов»

Листинг 1. «Пузырьковая сортировка»

Задание:

Упорядочить элементы массива по возрастанию с помощью «пузырьковой сортировки», известной также как «сортировка обменами».

Листинг:

```
program Array_sort;

const
n = 20;

var
  arr: array [1..n] of integer;

procedure array_fill;
var k : byte;
begin
    randomize;
    for k := 1 to n do arr[k] := 10 + random(90);
end;

procedure array_print;
var k : byte;
begin
    for k := 1 to n do write(arr[k], '   ');
    writeln;
end;

procedure array_bubble_sort;
var a, b, buf : integer;
begin
    for a := 1 to n - 1 do
       for b := 1 to n - a do
          if arr[b] > arr[b+1] then
          begin
            buf := arr[b];
            arr[b] := arr[b+1];
            arr[b+1] := buf;
         end;
end;
begin
    array_fill;
    array_print;
    array_bubble_sort;
    array_print;
end.
```

Тестирование программы:

```
18  75  61  39  79  58  99  30  92  82  59  94  68  44  80  48  62  16  15  94
15  16  18  30  39  44  48  58  59  61  62  68  75  79  80  82  92  94  94  99
```

Листинг 2. «Сортировка выделением»

Задание:

Упорядочить элементы массива по возрастанию с помощью «сортировки выделением», известной также как «сортировка выбором».

Листинг:

```
program Array_sort;
const
     n = 20;
var
  arr: array [1..n] of integer;
procedure array_fill;
var k : byte;
begin
    randomize;
    for k := 1 to n do arr[k] := 10 + random(90);
end;
procedure array_print;
var k : byte;
begin
    for k := 1 to n do write(arr[k], ' ');
    writeln;
end;
procedure array_selection_sort;
var a, b, max, mx: integer;
begin
   for b := n downto 2 do
   begin
      max := arr[1];
      mx := 1;
      for a := 2 to b do
         if arr[a] > max then begin
            max := arr[a];
            mx := a
         end;
      arr[mx] := arr[b];
      arr[b] := max;
   end;
end;

begin
    array_fill;
    array_print;
    array_selection_sort;
    array_print;
end.
```

Тестирование программы:

```
50  50  65  24  94  80  27  97  69  22  94  26  46  86  91  66  65  22  23  23
22  22  23  23  24  26  27  46  50  50  65  65  66  69  80  86  91  94  94  97
```

«Быстрая сортировка»

Листинг 1. «Быстрая сортировка»

Задание:

Упорядочить элементы массива по возрастанию методом «быстрой сортировки».

Листинг:

```
program Quick_sort;

const n = 20;

var   arr: array [1..n] of integer;

procedure array_fill;
var k : byte;
begin
    randomize; for k := 1 to n do arr[k] := 10 + random(90);
end;

procedure array_print;
var k : byte;
begin
    for k := 1 to n do write(arr[k], '  '); writeln;
end;

procedure array_quick_sort(a, b: integer);
var m, k, c, buf: integer;
begin
    m := a;
    k := b;
    c := arr[(m + k) div 2];
    repeat
        while arr[m] < c do inc(m);
        while arr[k] > c do dec(k);
        if m <= k then
        begin
            buf := arr[m];
            arr[m] := arr[k];
            arr[k] := buf;
            inc(m);
            dec(k);
        end;
    until m > k;
    if a < k then array_quick_sort(a, k);
    if m < b then array_quick_sort(m, b);
end;
begin
    array_fill;
    array_print;
    array_quick_sort(1, n);
    array_print;
end.
```

Тестирование программы:

```
42  59  33  31  61  27  89  93  49  16  76  24  42  75  60  91  10  50  45  86
10  16  24  27  31  33  42  42  45  49  50  59  60  61  75  76  86  89  91  93
```

«Двухмерные массивы»

Листинг 1. «Шахматная доска»

Задание:

Пользователь вводит адрес шахматного поля. Программа сообщает цвет.

Вводные данные:строка, представляющая из себя адрес шахматного поля.

Результирующие данные:«Цвет ячейки: Тёмный»/ «Цвет ячейки: Светлый»/«Это не адрес шахматной ячейки!»

Листинг:

```
program Chess_board;

var
  square: array ['a'..'h', 1..8] of string;
  v, i: char;
  h, j, i2: byte;
  a: string;

begin

   for i := 'a' to 'h' do
      for j := 1 to 8 do
      begin
         i2 := ord(i) - 64;
         if (i2 mod 2 = j mod 2) then
           square[i, j] := 'Тёмный'
         else
           square[i, j] := 'Светлый'
      end;

   write('Укажитеадресклетки: '); readln(a);

   if (length(a) = 2) then
     begin
        v := a[1]; h := ord(a[2]) - 48;
        if ((v >= 'a') and (v <= 'h') and (h >= 1) and (h <= 8)) then
          writeln('Цветячейки: ', square[v, h])
        else
writeln('Это не адрес шахматной ячейки!');
end
else
writeln('Это не адрес шахматной ячейки!');
end.
```

Тестирование программы:

```
Укажитеадресклетки:e5
Цвет ячейки: Тёмный
```

```
Укажитеадресклетки:h7
Цвет ячейки: Светлый
```

```
Укажитеадресклетки: k7
Это не адрес шахматной ячейки!
```

«Файлы»

Листинг 1. «Абецедарий»

Задание:

Из каждой строки исходного текстового файла в результирующий файл перенести первый символ.

Вводные данные:содержимое файла abecedarium.txt.

Результирующие данные:результирующий файл result.txtпереименованный в alphabet.txt.

Листинг:

```
program abecedarium;

var  f_original, f_result : text;s : string;

begin
   assign(f_original, 'C:/files/abecedarium.txt');
   reset(f_original);

   assign(f_result, 'C:/files/result.txt');
{append(f_result);}rewrite(f_result);

   while not eof(f_original) do
   begin
      readln(f_original, s);
      writeln(f_result, s[1]);
   end;

   close(f_original);
   close(f_result);

   rename(f_result, 'C:/files/alphabet.txt');
   erase(f_original);
end.
```

Тестирование программы:

abecedarium.txt	***result.txt →alphabet.txt***
Я ящерка	Я
ютящейся	ю
эпохи,	э
щемящий	щ
шелест	ш
чувственных	ч
цикад,	ц
хлопушка	х
фокусов	ф
убогих,	у
тревожный	т
свист,	с
рывок	р
поверх	п

оград.	о
Наитие,	Н
минута	м
ликованья,	л
келейника	к
исповедальня.	и
Земная	З
жизнь	ж
еще	е
дарит,	д
горя,	г
высокое	в
блаженство	б
алтаря.	а

Листинг 2. «Типизированные файлы»

Задание:

Продемонстрировать действие функций и процедур для работы с типизированными файлами: fileSize, filePos, seek, truncate.

Листинг:

```
program work_with_the_file;
var  file_string : file of string;
size, position, mediana : integer;
begin
   assign(file_string, 'C:/files/text.txt');
   reset(file_string);

   size := fileSize(file_string);
   writeln('Начальныйразмерфайла: ', size);

position := filePos(file_string);
   writeln('Начальноеположениеуказателя: ', position);

   seek(file_string, size);
   position := filePos(file_string);
   writeln('Теперь положение указателя – в конце файла: ', position);

seek(file_string, 0);
   position := filePos(file_string);
   writeln('Положение указателя - снова в начале файла: ', position);

mediana := size div 2;
   seek(file_string, mediana);
   position := filePos(file_string);
   writeln('Положениеуказателя – всерединефайла: ', position);

   truncate(file_string);
   size := fileSize(file_string);
   writeln('Итоговый размер файла после обрезки: ', size);
writeln('Итоговое положение указателя: ', position);
close(file_string);
end.
```

Тестирование программы:

Начальный размер файла: 3
Начальноеположениеуказателя: 0
Теперь положение указателя – в конце файла: 3
Положение указателя - снова в начале файла: 0
Положениеуказателя – всерединефайла: 1
Итоговый размер файла после обрезки: 1
Итоговое положение указателя: 1

text.txt до выполнения программы	***text.txtпосле выполнения программы***
Файл (англ. file) — блок информации на запоминающем устройстве компьютера, имеющий определённое логическое представление (начиная от простой последовательности битов или байтов и заканчивая объектом сложной СУБД), соответствующие ему операции чтения-записи (см. ниже) и, как правило, фиксированное имя (символьное или числовое), позволяющее получить доступ к этому файлу и отличить его от других файлов (см. ниже). Работа с файлами реализуется средствами операционных систем. Многие операционные системы приравнивают к файлам и обрабатывают сходным образом и другие ресурсы: - области данных (необязательно на диске); - устройства — как физические, например, порты или принтеры, так и виртуальные (генератор случайных чисел); - потоки данных (именованный канал); - сетевые ресурсы, сокеты; - прочие объекты операционной системы.	Файл (англ. file) — блок информации на запоминающем устройстве компьютера, имеющий определённое логическое представление (начиная от простой последовательности битов или байтов и заканчивая объектом сложной СУБД), соответствующие ему операции чтения-записи (см. ниже) и, как правило, фиксированное имя (символьное или числовое), позволяющее получить доступ к этому файлу и отличить его от других файлов (см. ниже).

«Файлы». Опорный конспект

Текстовые файлы

```
var
    file_text: text;
```

Типизированныефайлы

```
var

    file_numbers:file of integer;
    file_symbols:file of char;
    file_string:file of string;
```

Нетипизированныефайлы

```
var

    file_unknown:file;
```

Функции и процедуры для обработки файлов

assign(*file_name*, *file_path*) → привязка переменной *file_name*к файлу, находящемуся по адресу *file_path*.
reset(*file_name*) → открытие файла, привязанного к переменной *file_name*для чтения.
append(*file_name*) → открытие файла, привязанного к переменной *file_name*для записи. Предыдущие данные в файле при записи будут удаляться.
rewrite(file_name)→ открытие файла, привязанного к переменной *file_name*для записи. Новые данные в при записи будут добавляться в конец файла.
eof(*file_name*) → логическая функция, выясняющая, достиг ли внутренний указатель конца файла.
readln(*file_name*, *str*) →построчное чтение данных из файла.
writeln(*file_name*, *str*) → построчнаязапись данных из файл.
close(*file_name*) → закрытиефайла.Если открытый для чтения или записи файл не закрыть, то файл нельзя будет удалять и переименовывать.
rename(*file_name*, *new_file_path*) →переименование файла или изменение его месторасположения..
erase(*file_name*)→ удалениефайла.
fileSize(*file_name*)→ получение текущего размера файла.
filePos(*file_name*)→ получение текущей позиции внутреннего указателя .
seek(*file_name*, *position*) → перемещение внутреннего указателя на новую позицию.
truncate(*file_name*) →обрезкафайла. Из файла удаляются все данные, начиная с текущей позиции внутреннего указателя и до конца файла.

«Интервалы и множества»

Листинг 1. «Интервалы»

Задание:

Продемонстрировать базовые возможности по работе с интервалами.

Листинг:

```
program user_type;

type
        week_day = (Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday);
        work_day = Monday.. Friday;

var
        Day, Previous_Day, Next_Day: Week_day;
        Day1, Day2 :  Work_day;
        D : 0..6;

begin

         (* Первыйтест *)
        Day1 := Monday;
         Day2 := Tuesday;
        writeln(Day1 > Day2);

         (* Второйтест *)
        for Day := Sunday to Saturday do
        Begin
                D := ord(day);
                writeln(D);
        End;

         (* Третийтест *)
        Day := Wednesday;
        writeln('Номер дня недели: ', ord(Day));
        Previous_Day := pred(Day);
        writeln('Номер предыдущего дня недели: ', ord(Previous_Day));
        Next_Day := succ(Day);
        writeln('Номер следующего дня недели: ', ord(Next_Day));

end.
```

Тестирование программы:

1-тест:	***2-тест:***	***3-тест:***
False	0 1 2 3 4 5 6	Номер дня недели: 3 Номер предыдущего дня недели: 2 Номер следующего дня недели: 4

Листинг 2. «Множества»

Задание:

Продемонстрировать базовые возможности по работе с множествами.

Листинг:

```
program Auto_and_Salon;

type
        Auto = (Audi, BMW, Chevrolet, Ferrari, Ford, Honda,
                 Jeep, Nissan, Opel, Peugeot, Porsche, Renault,
                Toyota, Volvo);
        AutoSalon = set of auto;

var
        Mark : Auto;
        Salon_American, Salon_Japan, Salon_Europe : AutoSalon;
        MegaSalon : AutoSalon;

begin

        Salon_American := [Chevrolet, Ford, Jeep, Porsche];
        Salon_Europe := [Audi, BMW, Ferrari, Opel, Peugeot,
        Porsche, Renault, Volvo];
        Salon_Japan := [Honda, Nissan, Toyota];

        Mark := BMW;
        writeln('Марка: BMW');
        writeln;

        writeln('Автосалон: "Конныйсамурай"');
        if Mark in Salon_Japan then
                writeln('В этом автосалоне есть такой бренд.')
        else
                writeln('В этом автосалоне такой бренд не продаётся!');
writeln;

         megaSalon := Salon_American + Salon_Europe + Salon_Japan;
        writeln('Автосалон: "СуперМегаГиперСалон"');
        if Mark in MegaSalon then
                writeln('В этом автосалоне есть такой бренд.')
        else
                writeln('В этом автосалоне такой бренд не продаётся!');

end.
```

Тестирование программы:

```
Марка: BMW

Автосалон: "Конныйсамурай"
В этом автосалоне такой бренд не продаётся!

Автосалон: "СуперМегаГиперСалон"
В этом автосалоне есть такой бренд.
```

«Интервалы и множества», «Работа с диапазонами». Опорный конспект

Ограниченный тип

```
type
    temperature = -100..100;
    alphabet = ‘a’..’z’;
var
    t1, t2 : temperature;
    litera : alphabet;
```

Перечисляемый тип

```
type
  {Тип 1} = (Значение1, Значение2, …, ЗначениеN);
  {Подтип типа 1} = (ЗначениеA..ЗначениеB); {Перечисляемый ограниченный тип!}
```

ord(x) → возвращает порядковый номер в типе.

pred(x) → возвращает значение перечисляемого типа у которого порядковый номер в типе меньше на единицу.

succ(x) → возвращает значение перечисляемого типа у которого порядковый номер в типе больше на единицу.

Множества

```
type
    Тип1 = (Элемент1, Элемент2, …, ЭлементN);
    Тип2 = n...m;
    Множество1 = setof Тип1;
    Множество2 = set ofТип2;
    Множество3 = set of byte;
    Множество4 = setofinteger; {неверно объявленное множество, компилятор выдаст ошибку}
    Множество5 = set of char;
var
    ЭкземплярМножества1 : Множество1;
    ЭкземплярМножества2 : Множество2;
```

in → оператор принадлежности к множеству.

«Работа с диапазонами»

Листинг 1. «Поделочные камни»

Задание:

Даны десять полудрагоценных пород и их химические формулы:

АквамаринBe$_3$ Al$_2$ Si$_6$ O$_{18}$
Лазурит Na$_6$ Ca$_2$ (Al Si O$_4$)$_6$ (S O$_4$, S, Cl)$_2$
Малахит Cu$_2$ (C O$_3$) (O H)$_2$
Нефрит Ca$_2$ (Mg, Fe)$_5$ [Si$_4$ O$_{11}$] $_2$ (O H)$_2$
Родонит (Mn++, Fe++, Mg, Ca) Si O$_3$
ФлюоритCa F$_2$
Хризолит (Mg, Fe)$_2$ Si O$_4$
Чароит (K, Ba, Sr) (Ca, Na)$_2$ [Si$_4$ O$_{10}$] (O H, F) · H$_2$ O
Янтарь C$_{10}$ H$_{16}$ O+ (H$_2$ S)
Яшма (SiO$_2$) (Al$_2$ O$_3$) (Fe$_2$ O$_3$) (Ca O)

Выписатьдля каждого минерала номера в таблице Менделеевахимических элементов, входящих в его состав.

Листинг:

```
program chemistry;

type
   element = (H, He, Li, Be, B, C, N, O, F, Ne, Na, Mg, Al, Si, P, S, Cl, Ar, K, Ca, Sc, Ti, V, Cr, Mn, Fe, Co, Ni,
Cu, Zn,
                  Ga, Ge,  Asmiy, Se, Br, Kr, Rb, Sr, Y, Zr, Nb, Mo, Tc, Ru, Rh, Pd, Ag, Cd, Indiy, Sn, Sb, Te, I,
Xe, Cs,
                  Ba, La, Ce, Pr, Nd, Pm, Sm, Eu, Gd, Tb, Dy, Ho, Er, Tm, Yb, Lu, Hf, Ta, W, Re, Os, Ir, Pt, Au,
Hg, Tl, Pb,
                  Bi, Po, At, Rn, Fr, Ra, Ac, Th, Pa, U, Np, Pu, Am, Cm, Bk, Cf, Es, Fm, Md, No, Lr, Rf, Db, Sg,
Bh, Hs,
                  Mt, Ds, Rg, Cn, Uut, Fl, Uup, Lv, Uus, Uuo);

   elements = set of element;

const
    count = 10;

var
   e : element;
   name_substance : array [1..count] of string;
   chemical_substance : array [1..count] of elements;
   mineral : 1..count;

begin

    name_substance[1] := 'Аквамарин';
    name_substance[2] := 'Лазурит';
    name_substance[3] := 'Малахит';
    name_substance[4] := 'Нефрит';
    name_substance[5] := 'Родонит';
    name_substance[6] := 'Флюорит';
```

```
    name_substance[7] := 'Хризолит';
    name_substance[8] := 'Чароит';
    name_substance[9] := 'Янтарь';
    name_substance[10] := 'Яшма';




chemical_substance[1] := [Be, Al, Si, O]; {Аквамарин}
    chemical_substance[2] := [Na, Ca, Al, Si, O, S, Cl]; {Лазурит}
    chemical_substance[3] := [Cu, C, O, H]; {Малахит}
    chemical_substance[4] := [Ca, Mg, Fe, Si, O, H]; {Нефрит}
    chemical_substance[5] := [Mn, Fe, Mg, Ca, Si, O]; {Родонит}
    chemical_substance[6] := [Ca, F]; {Флюорит}
    chemical_substance[7] := [Mg, Fe, Si, O];  {Хризолит}
    chemical_substance[8] := [K, Ba, Sr, Ca, Na, Si, O, H, F]; {Чароит}
    chemical_substance[9] := [C, H, O, S]; {Янтарь}
chemical_substance[10] := [Si, O, Al, Fe, Ca]; {Яшма}

    for mineral := 1 to count do
    begin
        writeln(name_substance[mineral]);
        writeln('Состав:');
        for e := H to Uuo do
           if e in chemical_substance[mineral] then
             writeln('Элемент №', ord(e) + 1);
writeln;
    end;

end.
```

Тестирование программы:

Аквамарин Состав: Элемент №4 Элемент №8 Элемент №13 Элемент №14	Малахит Состав: Элемент №1 Элемент №6 Элемент №8 Элемент №29	Родонит Состав: Элемент №8 Элемент №12 Элемент №14 Элемент №20 Элемент №25 Элемент №26	Хризолит Состав: Элемент №8 Элемент №12 Элемент №14 Элемент №26	Янтарь Состав: Элемент №1 Элемент №6 Элемент №8 Элемент №16
Лазурит Состав: Элемент №8 Элемент №11 Элемент №13 Элемент №14 Элемент №16 Элемент №17 Элемент №20	Нефрит Состав: Элемент №1 Элемент №8 Элемент №12 Элемент №14 Элемент №20 Элемент №26	Флюорит Состав: Элемент №9 Элемент №20	Чароит Состав: Элемент №1 Элемент №8 Элемент №9 Элемент №11 Элемент №14 Элемент №19 Элемент №20 Элемент №38 Элемент №56	Яшма Состав: Элемент №8 Элемент №13 Элемент №14 Элемент №20 Элемент №26