Массивы

9 класс

- Основные теоретические сведения
- Примеры решения задач

Основные теоретические сведения

- Описание массива
- Описание константного массива
- Ввод элементов массива с клавиатуры
- Задание элементам массива случайных значений
- Вывод элементов массива
- Поиск элементов с заданными свойствами
  - Линейный поиск
  - Поиск минимального (максимального) элемента

Описание массива

- Массив в языке Pascal ABC описывается так:

  имя_массива: ***array****[индекс1..индексN]* ***of*** *тип_элементов;*

- Имя массива (идентификатор) задается по тем же правилам, что и имена переменных других типов;
- Служебное слово ***array*** означает массив.
- В квадратных скобках задается диапазон индексов элементов массива:
  - сначала указывают индекс первого элемента *индекс1*, затем ставят две точки, после которых указывают индекс последнего элемента *индексN*.
  - Диапазон индексов определяет максимально возможное количество элементов в массиве – **размер массива**.
- Тип элементов может быть любым, например: числа, символы, строки, массивы.
- Примеры описаний

Примеры описаний

- Имя массива *A*, диапазон индексов от 1 до 25, размер массива 25 целых чисел.

```
var A: array[1..25] of integer ;
```

- Этот массив можно описать и так (задав диапазон константами *n1* и *n2*):

```
const n1=1; n2=25;
var A: array[n1..n2] of integer;
```

- Имя массива T, диапазон индексов от 20 до 31, размер массива 12 вещественных чисел

```
var T: array[20..31] of real;
```

Описание константного массива

- Значения элементов, которые не изменяются при работе программы (констант), можно задавать в разделе описаний.
- Например, массив из 7 простых чисел можно описать следующим образом:

```
const
  A: array[1..7] of integer =(2,3,5,7,11,13,17);
```

Ввод элементов массива с клавиатуры

- Для ввода значений с клавиатуры используются стандартные процедуры *read* или *readln*.
- Например, следующая программа присваивает значения, вводимые с клавиатуры, пяти элементам массива.

```
var A: array[1..5] of integer;
i: integer;
begin
   for i:=1 to 5 do
       read(A[i]) ; {ввод значений элементов}
end.
```

Ввод элементов массива с клавиатуры

Для того, что бы программа могла при каждом запуске работать с массивами, которые содержат разное количество элементов, ввод можно организовать следующим образом:

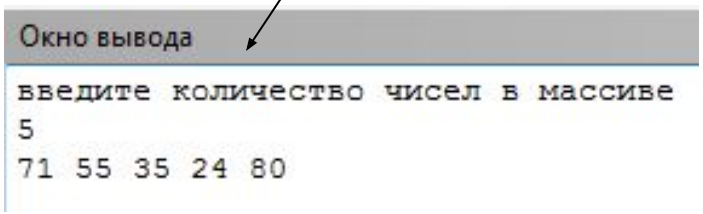
```
const n_max=100;
var A: array[1..n_max] of integer ;
i,n: integer;
begin
  writeln('введи количество элементов');
  readln(n);
  writeln('введи элементы');
  for i := 1 to n do
     read(A[i]) ; { ввод значений элементов }
end.
```

Задание элементам массива случайных значений

- Для задания элементам массива случайных значений используется функция *random(k)*, которая генерирует целое случайное число из промежутка [0; k-1].
- Случайным образом задать n элементов массива A. Значение каждого элемента – число из промежутка [0; 100].

```
var A: array[1..100] of integer;
      i,n:integer;
begin
  writeln('введите количество чисел в массиве');
  readln(n);
  for i:=1 to n do
  begin
    A[i]:=random(101);
    write(A[i],' ');
  end;
end.
```

Результат работы



Задание элементам массива случайных значений

- Если элементы массива должны принадлежать промежутку [a; b], то

```
A[i]:= random(b-a+1)+a;
```

- Для задания случайных значений вещественных чисел используется функция *random* без параметра, которая генерирует число из промежутка [0; 1).

```
A[i]:= random;
```

Вывод элементов массива

- Для вывода используют процедуры *write* или *writeln.* Процедура *write* выводит значения элементов массива в строку. При этом выводимые значения необходимо отделять пробелами или иными символами (например, запятой, точкой с запятой), иначе все они будут напечатаны слитно.
- Для вывода значений элементов в столбец используют процедуру *writeln.*

- Вывод n элементов массива в столбец - по одному в строке

```
for i:=1 to n do
    writeln(A[i]);
```

- Вывод n элементов массива в строку - через пробел
-

```
for i:=1 to n do
    write (A[i]);
```

Линейный поиск

- Самый простой способ поиска элементов массива с заданными свойствами – это последовательный просмотр всех элементов и проверка выполнения условий поиска.
- Такой алгоритм поиска называется **линейным** или **последовательным**.

Поиск минимального (максимального) элемента

- Минимальный элемент массива – элемент имеющий наименьшее значение среди всех элементов, а максимальный – наибольшее
- Для нахождение значения минимального (максимального) элемента массива нужно просмотреть все элементы массива и на каждом шаге сравнивать значение текущего элемента с уже найденным на предыдущих шагах значением минимума.

Поиск минимального (максимального) элемента

- Поиск максимального среди n элементов массива.

```
max:=A[1];
for i:=2 to N do
    if A[i]>max then
        max:=A[i];
```

- После завершения работы в переменной max будет храниться значение максимального элемента массива.

- Поиск минимального среди n элементов массива.

```
min:=A[1];
for i:=2 to N do
    if A[i]<min then
        min:=A[i];
```

- После завершения работы в переменной min будет храниться значение максимального элемента массива.

Основные типы задач

- Этапы решения задачи
- Нахождение суммы элементов массива
- Преобразование элементов массива
- Линейный поиск элемента с заданными свойствами
- Подсчет количества элементов с заданными свойствами
- Нахождение максимального (минимального) элемента массива

Этапы решения задач

- Определение исходных данных.
- Определение результатов.
- Составление алгоритма решения задачи.
- Определение типов переменных.
- Написание программы.
- Подготовка тестов и тестирование программы.

Нахождение суммы элементов массива

- Дан линейный массив из целых чисел. Найти сумму элементов линейного массива.

Определение исходных данных

- Переменная n - количество элементов в массиве,
- Переменная a – линейный массив.

Определение результатов

- Переменная *S* - сумма элементов массива

Алгоритм решения задачи

- Ввод исходных данных. Массив вводится поэлементно.
- Определение начального значения для суммы (*S*:=0).
- В цикле прибавляем очередной элемент массива к текущему значению переменной, хранящей сумму.
- Вывод результата.

Описание переменных

- Переменные *n* и *S* имеют тип *integer*
- Переменная *a* имеет тип *array*
- Элементы массива имеют тип *integer*.
- Для работы также необходима переменная *i* типа *integer* – счетчик цикла.

Программа

```
Var    n, i, S, P: integer;
    a: array[1..10] of integer;
Begin
    Write('введите количество элементов массива n=');
    Readln(n);
    Write('вводите элементы массива через пробел');
    For i:=1 to n do
        Read(a[i]);
    S:=0;
    For i:=1 to n do
        S:=S+a[i];
    writeln('сумма элементов = ',S);
End.
```

Тестирование

- Запустите программу и введите значения

```
n=5
3 2 4 -1 3
```

- Проверьте, результат должен быть следующим:

```
сумма элементов = 11
```

- Проверить правильность вычислений можно на калькуляторе.

Вопросы

- Что нужно изменить в программе, для того, чтобы вычислялось произведение элементов массива? Внесите изменения.
- Измените программу так, чтобы получить сумму и произведение за один проход по массиву.

Преобразование элементов массива

- Задан линейный массив. Преобразовать его элементы по следующему правилу: положительные увеличить на 5, а отрицательные элементы массива заменить их модулями.

Определение исходных данных и результатов

- Исходные данные
  - Переменная n - количество элементов в массиве,
  - Переменная a – линейный массив.
- Результат
  - Преобразованный массив *a*

Алгоритм решения задачи

- Ввод исходных данных. Массив вводится поэлементно.
- В цикле проверяем текущий элемент, если он положительный, то прибавляем к нему 5, если отрицательный - заменяем его модулем.
- Вывод результата.

Описание переменных

- Переменная *n* имеет тип *integer*
- Переменная *a* имеет тип *array*
- Элементы массива имеют тип *integer*.
- Для работы также необходима переменная *i* типа *integer* – счетчик цикла.

Программа:

```
Var n, i: integer;
    A: array[1..10] of integer;
Begin
    Write('введите количество элементов массива n=' );
    Readln(n);
    Writeln('вводите элементы массива через пробел' );
    For i:=1 to n do
     Read(A[i]);
    For i:=1 to n do
    begin
      if A[i]>0 then
       A[i]:=A[i]+5;
      if A[i]<0 then
       A[i]:=abs(A[i]);
    end;
    writeln('преобразованный массив' );
    For i:=1 to n do
      write(A[i], ' ')
End.
```

Тестирование

- Запустите программу и введите значения

```
n=5
3 -2 0 -1 5
```

- Проверьте, результат должен быть следующим:

```
преобразованный массив
8 2 0 1 10
```

Линейный поиск элемента с заданными свойствами

- В заданном линейном массиве определить есть ли хотя бы один элемент который является нечетным, не кратным 7 числом, если “да”, то напечатать его номер.

Определение исходных данных и результатов

- Исходные данные:
  - Переменная *n* - количество элементов в массиве,
  - Переменная *a* – линейный массив.
- Результат:
  - *Переменная k* – номер позиции, на которой находится элемент
  - Вывод «нет», если элемент не найден.

Алгоритм решения задачи

- Ввод исходных данных.
- Для решения задачи воспользуемся алгоритмом поиска с барьером.
  - Занесем элемент, удовлетворяющий условию задачи (например, число 5) на последнее место в массив, а затем будем просматривать элементы с начала.
  - Как только элемент найден, остановимся.
  - Если элемент найден на месте n+1, значит в исходном массиве нет элемента, удовлетворяющего условию задачи, иначе печатаем номер найденного элемента.
- Вывод результата.

Описание переменных

- Переменные *n u k* имеют тип *integer*
- Переменная *a* имеет тип *array*
- Элементы массива имеют тип *integer*.
- Для работы также необходима переменная *i* типа *integer* – счетчик цикла.

Программа:

```
Var     i,k,n: integer;
    a: array [1..20] of integer;
Begin
    Writeln('введите количество элементов в массиве');
    Readln(n);
    Writeln('введите элементы');
    For i:=1 to n do
        Read(a[i]);
    a[n+1]:=5;
    k:=1;
    while (a[k] mod 2=0) or (a[k] mod 7=0) do
        k:=k+1;
    if k=n+1 then
        writeln('в массиве нет таких элементов')
    else
        writeln('элемент ',a[k],' стоит на месте ',k);
End.
```

Тестирование

- Запустите программу и введите значение

```
n=5
2 7 21 3 4
```

- Проверьте, результат должен быть следующим:

```
элемент 3 стоит на месте 4
```

Вопросы

- Какой ответ выдаст программа, если в массиве несколько элементов, удовлетворяющих условию задачи? Почему?
- Что нужно изменить в программе, чтобы выдавался не первый из найденных элементов, а последний?
- Измените условие цикла *while*, так чтобы использовалась логическая операция *not*.

Подсчет количества элементов с заданными свойствами

- В заданном линейном массиве посчитать количество элементов, равных заданному числу x.

Определение исходных данных и результатов

- Исходные данные:
  - Переменная *n* - количество элементов в массиве,
  - Переменная *a* – линейный массив.
  - Переменная *x* – искомое число
- Результат:
  - Переменная *k* количество элементов;
  - Если *k*=0, то таких элементов в массиве нет.

Алгоритм решения задачи.

- Ввод исходных данных.
- Для решения задачи воспользуемся алгоритмом линейного поиска.
  - До просмотра элементов массива k=0.
  - Будем просматривать элементы с начала.
  - Как только элемент найден, увеличим значение k.
- Проверка значения k.
- Вывод результата.

Описание переменных

- Массив целых чисел (*a*),
- переменные типа *integer*
  - количество элементов (*n*),
  - искомое число (*x*),
  - количество найденных (*k*),
  - текущий элемент (*i*);

Программа

```
Var i,k,n,x: integer;
    a: array [1..20] of integer;
Begin
    Writeln('введите количество элементов в массиве' );
    Readln(n);
    Writeln('введите элементы');
    For i:=1 to n do
        Read(a[i]);
    Writeln('введите число x');
    Readln(x);
    k:=0;
    for i:=1 to n do
      if a[i]=x then
        k:=k+1;
    if k=0 then
        writeln('в массиве нет таких элементов' )
    else
        writeln('в массиве ',k,' элементов =',x);
End.
```

Тестирование

Запустите программу и введите значение

```
   n=5
введите элементы
   2 2 3 2 4
введите число x
   2
```

Проверьте, результат должен быть следующим:

```
в массиве 3 элементов =2
```

Вопросы

- Какие изменения нужно внести в программу, что бы находили не только количество элементов, равных x, но их сумму.
- Предложите разные способы.
- Определите, какой процент от общего количества составляют найденные элементы.

Нахождение максимального (минимального) элемента массива

- В заданном линейном массиве определить, какой из элементов минимальный или максимальный встречается раньше.

Определение исходных данных и результатов

- Исходные данные:
  - Переменная *n* - количество элементов в массиве,
  - Переменная *a* – линейный массив.
- Результат:
  - Текстовый ответ – минимальный встречается раньше или максимальный встречается раньше.

Алгоритм решения задачи.

- Ввод исходных данных.
- Для решения задачи воспользуемся
  - Найдем минимальный элемент массива (min) и его номер (nmin)
  - Найдем максимальный элемент массива (max) и его номер (nmax)
  - Сравним значения номера максимального и номера минимального элементов
- Вывод результата.

Описание переменных

- Переменная *n* имеет тип *integer*
- Переменная *a* имеет тип *array*
- Элементы массива имеют тип *integer*.
- Переменные *min*, *max, nmin*, *nmax* имеют тип *integer*.
- Для работы также необходима переменная *i* типа *integer* – счетчик цикла.

Программа

```
Var   i,k,n,nmin,nmax,min,max,r: integer;
      a: array [1..20] of integer;
Begin
      Writeln('введите количество элементов в массиве');
      Readln(n);
      Writeln('введите элементы');
      For i:=1 to n do
         Read(a[i]);
      {поиск минимального элемента}
      min:=a[1]; nmin:=1;
      for i:=2 to n do
         if a[i]<min then
         begin
            min:=a[i];
            nmin:=i;
         end;
      {поиск максимального элемента}
      max:=a[1]; nmax:=1;
      for i:=2 to n do
         if a[i]>max then
         begin
           max:=a[i];
           nmax:=i;
       end;
      if nmin<nmax then
        writeln('минимальный встретился раньше')
      else
        writeln('максимальный встретился раньше')
End.
```

Тестирование

- Запустите программу и введите значение

```
n=5
введите элементы
2 5 1 3 4
```

- Проверьте, результат должен быть следующим:

```
максимальный встретился раньше
```

Вопросы

- Какой результат выдаст программа, если ввести все элементы, равные 3? Почему?
- Внесите в программу изменения, чтобы ответ был верным.
- Можно ли получить ответ не храня значения минимального и максимального элементов в соответствующих переменных? Как это сделать?