

# Одномерные и двумерные массивы

Массив как структурированный тип данных;  
типичные алгоритмы обработки массивов;  
сортировка массивов; организация поиска  
элемента в массиве

# Цель урока

---

- Ознакомится с понятием массива как структурированного типа данных.
- Изучить типичные алгоритмы обработки одномерных и двумерных массивов.

# Проверка домашнего задания

---

- Какие циклы без параметра вы знаете? Опишите их синтаксис.
- Проиллюстрируйте семантику работы каждого оператора цикла без параметра при помощи блок-схемы.
- Опишите синтаксис циклического оператора с параметром. Приведите пример его использования.

# Повторение изученного материала

---

1. Составьте программу для нахождения  $n$ -го члена арифметической прогрессии.
2. Составьте программу для нахождения суммы введенных  $n$  последовательных чисел.

# Массив как структурированный тип данных

---

- Структурированные типы данных базируются на простых (скалярных) типах данных и могут содержать их различные комбинации.
- К структурированным типам относятся: *тип массив, тип запись, тип множество, файловый тип.*

# Массив как структурированный тип данных

---

*Массив* – это структура данных, которая представляет собой однородную, фиксированную по размеру и конфигурации совокупность элементов простой или составной структуры, упорядоченная по номерам.

# Массив как структурированный тип данных

---

Массив определяется именем (идентификатором) и количеством размерностей (координат), необходимых для указания местонахождения требуемого элемента массива. В зависимости от количества размерностей выделяют одномерные, двумерные и т.д. элементы массива.

# Массив как структурированный тип данных

---

Описание массива:

`array[тип индекса] of <базовый тип>.`

Тип индекса – любой порядковый тип.

Массив можно описывать непосредственно в разделе описания переменных.



# Массив как структурированный тип данных

---

Пример одномерного массива:

7	-8	9.1	8	1.1	0.9	-2	-5	9	-7	2.3	5.4	6.7	90	10	-2	1.9	1.2	2.3	4
a[4]										a[14]									

# Массив как структурированный тип данных

Пример двумерного массива:

7	-9	1.1	$a[1,3]$
2.1	-8	-9	
-7	-4	1.3	$a[3,2]$
3.4	5.6	0	

# Массив как структурированный тип данных

---

- Примеры описания массивов:

для одномерного массива:

```
const n=20;
```

```
var a:array[1..n] of real;
```

для двумерного массива:

```
const n=4, m=3;
```

```
var a:array[1..n,1..m] of real;
```

# Массив как структурированный тип данных

---

- Другой способ – массив сначала описывается в разделе описания типов, а в разделе описания переменных необходимые переменные объявляются как ранее определенный тип

# Массив как структурированный тип данных

---

Примеры описания массивов:

для одномерного массива:

```
const n=20;  
type mas=array[1..n] of real;  
var a,b:mas;
```

для двумерного массива:

```
const n=4,m=3;  
type mas=array[1..n,1..m] of real;  
var a,b:mas;
```

# Массив как структурированный тип данных

---

- Обращение к конкретному элементу массива производится при помощи указания массива и номера необходимого элемента. Например: `a[7]`, `b[4,5]`.
- Ввод и вывод элементов массива осуществляется при помощи циклов.

# Массив как структурированный тип данных

---

- Пример ввода значений в одномерный массив:

```
For i:=1 to n do  
  read(a[i]);
```

- Пример вывода значений одномерного массива:

```
For i:=1 to n do  
  write(a[i]);
```

# Массив как структурированный тип данных

---

- Функция `Random` возвращает случайное число:

`Random`<максимальное число>.

Результат – случайное число в диапазоне `[0; максимальное число)`.

Перед использованием функции `Random` используют процедуру `Randomize`



# Типичные алгоритмы обработки массивов

- Поиск наибольшего (наименьшего) элемента массива и определение его индекса:

```
max:=a[1]; nom:=1;
```

```
for i:=1 to n do
```

```
  if a[i]>max then
```

```
    begin
```

```
      max:=a[i];
```

```
      nom:=i
```

```
    end;
```

# Типичные алгоритмы обработки массивов

---

- Вычисление суммы (произведения) элементов массива:

```
s:=0;
```

```
for i:=1 to n do
```

```
  s:=s+a[i];
```

# Типичные алгоритмы обработки массивов

---

- Вычисление суммы (произведения) элементов массива, удовлетворяющих заданному условию:

```
for i:=1 to n do
```

```
if a[i] mod 2=0 then s:=s+a[i];
```

# Типичные алгоритмы обработки массивов

---

- Подсчет количества элементов массива, удовлетворяющих заданному условию:

```
kol:=0;
```

```
for i:=1 to n do
```

```
    if  $a[i] \text{ div } 2 = a[i]/5$  then kol:=kol+1;
```

# Сортировка массивов

---

- Сортировка массивов – упорядочивание элементов массива по убыванию или возрастанию.

# Сортировка массивов

---

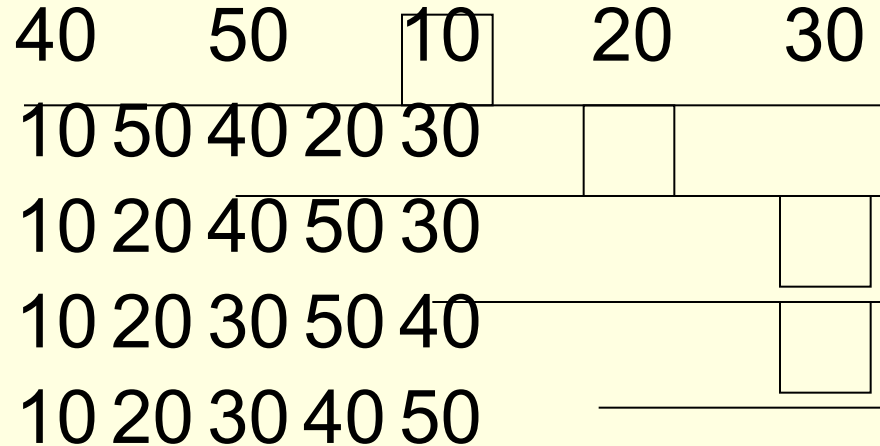
- Алгоритм сортировки выбором.

Пусть задан массив: 40, 50, 10, 20, 30.

1. Установить номер наименьшего элемента массива.
2. Поменять местами наименьший и первый элемент.
3. Не принимая во внимание первый элемент, выполнить пункты 1 и 2 над оставшейся частью массива (без первого элемента).
4. Пункт 3 повторять пока остаток массива не сократится до одного элемента.

# Сортировка массивов

- Алгоритм сортировки выбором.



Для выполнения алгоритма потребуется  $n-1$  проход, где  $n$  – количество элементов.

# Сортировка массивов

---

- Алгоритм сортировки обменом (пузырьковая сортировка).

Слева направо поочередно сравниваются два соседних элемента, и если взаиморасположение не соответствует условию упорядоченности, то они меняются местами. Далее два следующих соседних элемента и т.д. до конца массива.



# Сортировка массивов

---

- Алгоритм сортировки обменом (пузырьковая сортировка).

После первого такого прохода на последней  $n$ -ой позиции массива будет стоять максимальный элемент («всплыл» пузырек). Т.к. максимальный элемент уже стоит на своей позиции, то второй проход обменов выполняется до  $n-1$  элемент, и т.д. Всего потребуется  $n-1$  проход.

# Сортировка массивов

---

- Алгоритм сортировки вставкой.

Массив разделяется на две части – отсортированную и неотсортированную. Элементы из неотсортированной части последовательно выбираются и вставляются в отсортированную часть так, чтобы не нарушить в ней упорядоченность.

# Сортировка массивов

---

- Алгоритм сортировки вставкой.

В начале работы алгоритма в качестве отсортированной части массива принимают только один первый элемент. Всего требуется  $n-1$  проход.

# Организация поиска элементов в массиве.

---

- **Последовательный поиск.**

Элементы массива просматриваются поочередно, и каждый из них сравнивается со значением, которое необходимо найти.

# Организация поиска элементов в массиве.

- Последовательный поиск.

```
Program poisk;
Const n=20;
Var a:array[1..n] of integer;
    x,i:integer;
    flag:boolean;
Begin
Clrscr;
Randomize;
for i:=1 to n do
begin
a[i]:=random(10);
writeln(a[i]);
end;
write('vvedite iskomoe znachenie');
readln(x);
flag:=false;
for i:=1 to n do
if a[i]=x then
begin
flaf:=true;
writeln(i);
end;
if flag=false then writeln('iskomogo elementa v massive net');
Readkey
End.
```

# Организация поиска элементов в массиве.

---

- Двоичный поиск. Применяется только в упорядоченном массиве. Исходный массив делится пополам и для сравнения выбирается средний элемент. Если он совпадает с искомым, то поиск заканчивается. Если же средний элемент меньше искомого, то все элементы левее его также будут меньше искомого. Следовательно, их можно исключить из зоны дальнейшего поиска, оставив только правую часть массива.

# Организация поиска элементов в массиве.

---

- Аналогично, если средний элемент больше исходного, то отбрасывается правая часть, а остается левая.

На втором этапе выполняются действия над оставшейся половиной массива. В результате после второго этапа остается  $\frac{1}{4}$  часть массива.

И т.д., пока или элемент будет найден, или зона поиска станет равной нулю. В последнем случае искомого элемента в массиве нет.

# Домашнее задание

---

- Какие структурированные типы вы знаете?
- Что представляет собой массив, как структура данных?
- Какие типы данных могут выступать в качестве индексов и элементов массива?
- Как осуществляется вывод значений элементов массива?
- Как заполнить элементы массива случайными значениями?
- Приведите пример задач, иллюстрирующих типичные алгоритмы обработки массивов.
- Какие способы сортировки массивов вы знаете?
- В чем заключается принцип последовательного и двоичного поиска?



# Домашнее задание

---

- Вычислить сумму и произведение элементов массива (массив задается случайными числами).
- Создайте массив, содержащий только положительные элементы исходного массива (массив задается случайными числами).
- Посчитайте сумму цифр в числе.