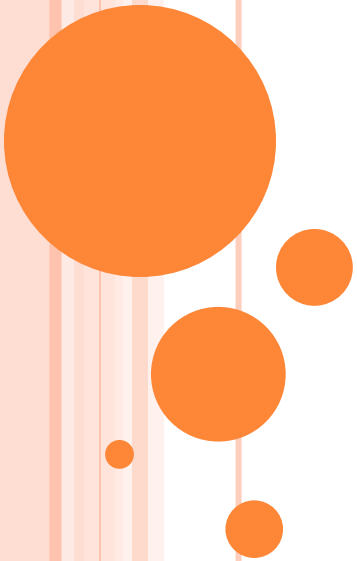


Массивы

Лекция №4

План лекции

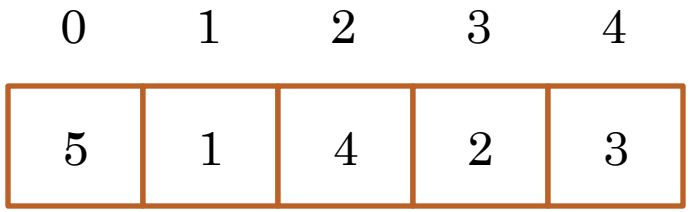
- Что такое массив?
- Как работают с массивом?
- Операции над массивами без перестановки элементов
- Операции над массивами с перестановкой элементов

Исходные предпосылки

- Необходимость обработки больших объемов однотипных данных
- Необходимость прямого доступа к отдельным элементам
- Оперативная память похожа на одномерный массив
  - ячейки (элементы) расположены друг за другом
  - каждая ячейка имеет свой адрес (номер)

Массив

- Пронумерованная последовательность элементов одного и того же типа
  - Элементы одного и того же типа
  - Элементы расположены последовательно
  - Конечное количество элементов
  - Каждый элемент имеет свой номер
  - Элементы нумеруются с 0 до n-1, если всего элементов - n



Формат описания массива

- тип имя[кол_во_эл]={знач1,знач2,знач3};
  - Количество элементов – целая константа
  - Номера элементов – целые
  - Границы контролирует программист!!!
- //Примеры описания массивов
- int A[5]={2,1,5,4,3};
- float B[3]={1.12, 2.25, 0.18};
- char C[4]={‘A’,’b’,’1’,’!’};

- int A1[10];//значения элементов - случайные
- float B1[7];
- char C1[5];

Примеры работы с массивом

```
int X[5]={2,1,4,5,3}, k, m, n,i=3;
k=X[0];
X[0]=X[4];
X[4]=k;
m=(X[1]+X[3])/2;
n=X[i]-X[i-1];
i++;
k=X[i];
printf("X[0]= %d X[4] = %d\n", X[0], X[4]);
printf("k= %d m = %d n=%d\n", k, m, n);

k=X[5];//Выход за границу массива!!!
i=0;
m=X[i]-X[i-1];//Будет ли выход за границу?
```

Особенности обработки массивов

- Для выполнения одних и тех же действий над всеми элементами массива используют циклы
- Для анализа свойств элементов массива используют условные переходы
- Недопустимы выходы за границы массивов

Вывод на экран элементов массива

```
#include<stdio.h>
void main(void)
{
int X[5]={2,1,4,5,3},n=5,i;
for(i=0;i<n;i++)
   printf("X[%d]=%d\n", i, X[i]);
}
```

Ввод с клавиатуры элементов массива

```
#include<stdio.h>
void main(void)
{
int X[5],n=5,i;
for(i=0;i<n;i++)
   {
    printf(“X[%d]=”,i]);
    scanf(“%d”, &X[i]);
   }
}
```

Операции над массивами без перестановок

- Сумма элементов массива
- Сумма элементов массива, соответствующих признаку
- Подсчет элементов массива, соответствующих признаку
- Нахождение минимума или максимума
- Поиск элемента, соответствующего признаку

Операции над массивами с перестановками

- Удаление элемента из массива
- Добавление элемента в массив
- Изменение порядка следования элементов на противоположный
- Упорядочение по возрастанию или убыванию

Сумма элементов массива

```
#include<stdio.h>
void main(void)
{
int X[5]={2,1,4,5,3},n=5,I,s=0;
for(i=0;i<n;i++)
   printf("X[%d]=%d\n", i, X[i]);
for(i=0;i<=n-1;i++)
   s+=X[i]; //накопление суммы s=s+X[i];

printf("s=%d\n", s);
}
```

Сумма положительных элементов массива

```
#include<stdio.h>
void main(void)
{
int X[5]={2,-1,4,-5,3},n=5,i,s=0;
for(i=0;i<5;i++)
   printf("X[%d]=%d\n", i, X[i]);
for(i=0;i<=4;i++)
   if(X[i]>0)
         s+=X[i]; //накопление суммы s=s+X[i];

printf("Sum of positive numbers s=%d\n", s);
}
```

Количество отрицательных элементов массива

```
#include<stdio.h>
void main(void)
{
int X[5]={2,-1,4,-5,3},n=5,i,q=0;
for(i=0;i<n;i++)
   printf("X[%d]=%d\n", i, X[i]);
for(i=0;i<=n-1;i++)
   if(X[i]<0)
         q++; //подсчет количества элементов

printf("Num of negative numbers q=%d\n", q);
}
```

Поиск минимального элемента

```
#include<stdio.h>
void main(void)
{
int X[5]={2,-1,4,-5,3},n=5,i,imin=0,min;
for(i=1;i<=n-1;i++)
   if(X[i]<X[imin])
         imin=i; //поиск номера мин. элемента

printf("Number of min =%d\n", imin);

min=?????//Как получить значение мин. эл.
//Как найти максимальный элемент?
}
```

Поиск элемента с заданным значением

```
#include<stdio.h>
void main(void)
{
int X[5]={2,4,5,4,3},n=5,i,v=4,k;
k=-1;
for(i=0;i<=4;i++)
    if(X[i]==v)
            {
            k=i; //сохраняем номер найденного эл-та
             break;//??? Что будет, если удалить break?
            }
if(k!=-1)
    printf("Value %d has number %d\n", v, k);
else
    printf ("Value %d is not found!!!", v);
}
//??? Найден первый или последний повторяющийся элемент?
```

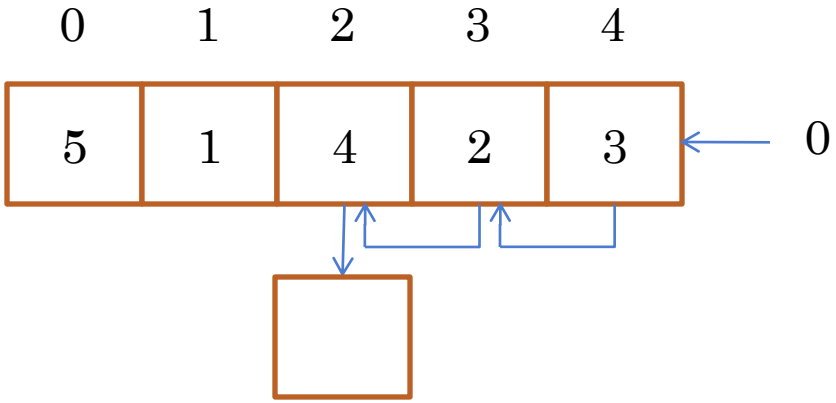
Поиск элемента по заданному признаку

```
#include<stdio.h>
void main(void)
{
int X[10]={2,4,5,4,3,2,1,6,3,1},n=10,I,k;
k=-1;
for(i=1;i<n-1;i++)//Почему такие параметры цикла?
    if(X[i]>X[i-1] && X[i]>X[i+1])// К чему приведет изменение знаков?
            {
            k=i; //сохраняем номер найденного эл-та
             break;//??? Что будет, если удалить break?
            }
if(k!=-1)
    printf("Value has number X[%d] =%d is …\n", k, X[k]);
else
    printf ("…  not found!!!");
}
//??? Что за элемент мы ищем?
```

Поиск НОД (фрагмент)

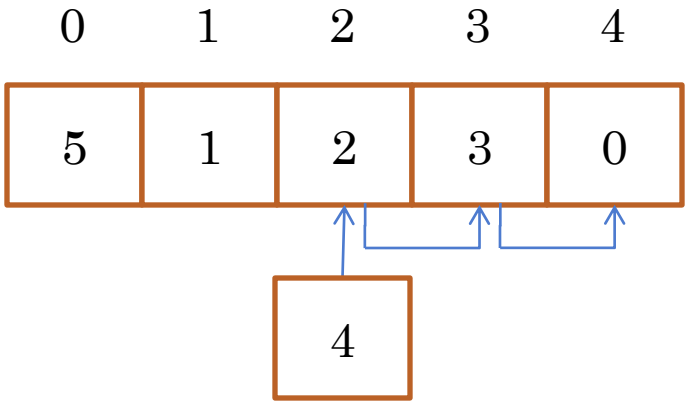
- for(i=0;i<n;i++)
  - for(j=min;j>0;j--)
    - {if(X[i]%j!=0)
      - break;
    - if(i==n)
      - break;
    - }

Удаление элемента из массива




```
//Фрагмент программы
k=2;
n=5; n_busy=n; n_free=0;
v=X[k];
for(i=k;i<n-1;i++)//сдвигаем к началу, заполняя «дыру»
    X[i]=X[i+1];
X[i]=0;//Последний элемент освободился
n_busy--;n_free++;
```

Добавление элемента в массив




```
//Фрагмент программы
n=5;
for(i=n-1;i>k;i--)
   X[i]=X[i-1];//сдвигаем к концу, освобождая место
X[?]=v;//что должно быть вместо ?
n_busy++; n_free--;
```

Упорядочение по возрастанию

```
//Фрагмент программы
int X[5]={2,1,5,4,3},i,j,tmp,n=5;

for(i=0;i<n;i++)//вложенный цикл -
   универсальность
   for(j=0;j<n-1,j++)
         if(x[j]>x[j+1]
               {tmp=x[j];//тяжелые - вниз
               x[j]=x[j+1];//легкие - вверх
               x[j]=tmp;}
```

Упорядочение по убыванию

```
//Фрагмент программы
int X[5]={2,1,5,4,3},i,j,jmax,tmp,n=5;

for(i=0;i<n;i++)
   {
   jmax=i;
   for(j=i;j<n-1;j++)
          if(x[j]>x[jmax]
               jmax=j;
   tmp=x[i];//упорядоченная часть растет
   x[i]=x[jmax];//неупорядоченная - укорачивается
   x[jmax]=tmp;
   }
```

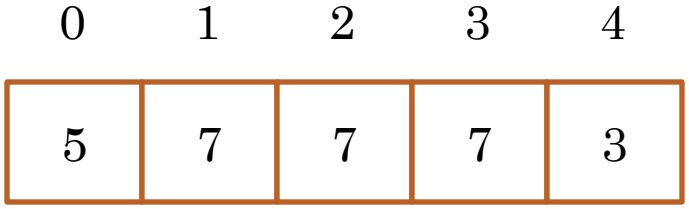
Копирование массивов

```
//Фрагмент программы

int X[5]={1,-2,3,-4,5}, Y[5],i;
for(i=0;i<5;i++)
   Y[i]=X[i];
```

Копирование элементов по признаку

```
//Фрагмент программы


int X[5]={1,-2,3,-4,5}, Y[5],i,j=0;
for(i=0;i<5;i++)
   if(X[i]>0)
          {Y[j]=X[i];
           j++;}
if(j==0)
   printf("Not found");
```

Поиск последовательности из одинаковых чисел



- Внешний цикл – от начала до предпоследнего i=0..n-2
- Внутренний цикл – от следующего после текущего до последнего j=i+1…n-1
  - Если X[i]==X[j] И (? ИЛИ ?)
    - сохраняем номер начала k=i;
  - Если X[i]==X[j] И X[j]!=X[j+1]
    - сохраняем номер конца

Реализация (Фрагмент программы)

```
for(i=0;i<n-1;i++)
        for(j=i+1;j<n-1;j++)
                if(X[j]==X[i]&&(i==0||X[i]!=X[i-1]))
                {
                        if(j==i+1)
                        {
                                k=i;
                                m=j;
                        }
                        else
                                if(X[j]!=X[j+1])
                                {
                                        k1=k;
                                        m1=m;
                                }
                                else
                                        m=m+1;
                }
```

Удаление элемента, равного среднему его соседей (Фрагмент программы)

```
for(i=0;i<n;i++)
        cout<<"X["<<i<<"]="<<X[i]<<endl;
for(i=0;i<n-2;i++)
{
        sr=(X[i-1]+X[i+1])/2;
        if(X[i]==sr)
        {
                k=i;
                for(j=k;j<n-1;j++)
                        X[j]=X[j+1];
                n--;
                Y[m]=k;
                Z[m]=X[k];
                m++;

        }

}
```

Заключение

- Массив позволяет обрабатывать упорядоченные последовательности данные
- Возможен прямой доступ к элементам по номеру
- Для обработки массивов требуются циклы в сочетании с условными переходами
- Обычно переменная цикла связана с индексами элементом массива
- Параметры циклов выбирают так, чтобы не допустить выхода за границу массива