آرایه‌ها ۱

مبانی برنامه‌نویسی

(۱۳۹۹-۱۳-۱۱)

جلسه‌ی بیست‌ویکم

دانشگاه شهید بهشتی

پاییز ۱۳۹۹

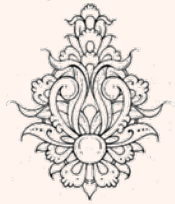
دانشکده‌ی مهندسی برق و کامپیوتر

احمد محمودی ازناوه

http://faculties.sbu.ac.ir/~a_mahmoudi/

فهرست مطالب

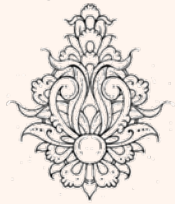
- تعریف آرایه
- نکاتی در مورد آرایه‌ها
- آرایه‌های موازی



مسأله‌ی کلاسیک هانوی (ادامه...)

```
void hanoi ( int nDisk, char start, char temp, char finish, bool flg){
    static int itr;
    if(flg)
        itr=0;
    if ( nDisk == 1 ){
        cout << setw(4)<<++itr<<") "<< start << " --> " << finish << endl;
    }
    else{
        hanoi ( nDisk - 1, start, finish, temp );
        cout <<setw(4)<<++itr<<") "<<start << " --> " << finish << endl;
        hanoi ( nDisk - 1, temp, start, finish );
    }
}
```

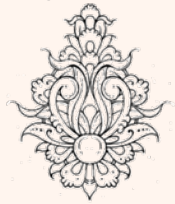
```
#define FIRST_CALL true
void hanoi ( int nDisk, char start, char temp, char finish,bool flg=false);
int main(){
    hanoi(3, 'A', 'B', 'C', FIRST_CALL);
    cout<<endl;
    hanoi(5, 'A', 'B', 'C',FIRST_CALL);
    return 0;
}
```



مسأله‌ی کلاسیک هانوی (ادامه...)

```
void hanoi ( int nDisk, char start, char temp, char finish,int num){
    static int itr;
    itr=num;
    if ( nDisk == 1 ){
        cout << setw(4)<<++itr<<") "<< start << " --> " << finish << endl;
    }
    else{
        hanoi ( nDisk - 1, start, finish, temp, itr );
        cout <<setw(4)<<++itr<<") "<<start << " --> " << finish << endl;
        hanoi ( nDisk - 1, temp, start, finish, itr );
    }
}
```

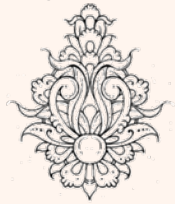
```
void hanoi ( int nDisk, char start, char temp, char finish,int num=0);
int main(){
    hanoi(3, 'A', 'B', 'C');
    cout<<endl;
    hanoi(5, 'A', 'B', 'C');
    return 0;
}
```



معایب و مزایای توابع بازگشتی

- پیاده‌سازی با دستورالعمل‌های کم‌تر همراه با ظرافت و خوانایی بهتر برنامه

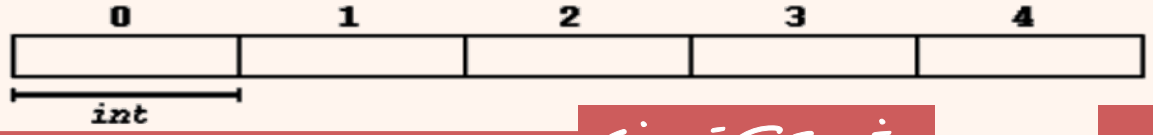
- Overhead بالا به دلیل فراخوانی تابع
- نگاه‌داشتن آدرس بازگشتی، ایجاد متغیرها
- مدیریت حافظه
  - در هر بار فراخوانی تابع مقداری حافظه اختصاص داده شده مصرف می‌شود.



آرایه‌ها

آرایه‌ها

- گروهی از محل‌های متوالی حافظه که دارای نام و نوع یکسانی هستند.
- برای مراجعه به محل یا عنصری خاص می‌باید نام آرایه و شماره‌ی عنصر مورد نظر مشخص شود.



شیوه‌ی تعریف

```
type name [elements];
```

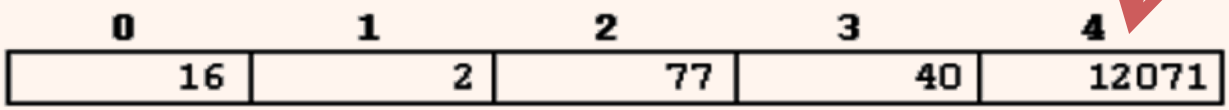
شیوه‌ی دسترسی

```
name[index];
```

```
int A[5] = {16, 2, 77, 40, 12071};
```

```
int A[] = {16, 2, 77, 40, 12071};
```

A[4]



مثال

```
#include <iostream>
using namespace std;
int main (){
    const int SIZE = 7;
    int numbers[SIZE] = {1, 2, 4, 8};
    cout << "Here are the contents of the array:\n";
    for (int index = 0; index < SIZE; index++)
        cout << numbers[index] << " ";
    cout << endl;
    return 0;
}
```

```
Here are the contents of the array:
1 2 4 8 0 0 0
```

مثال

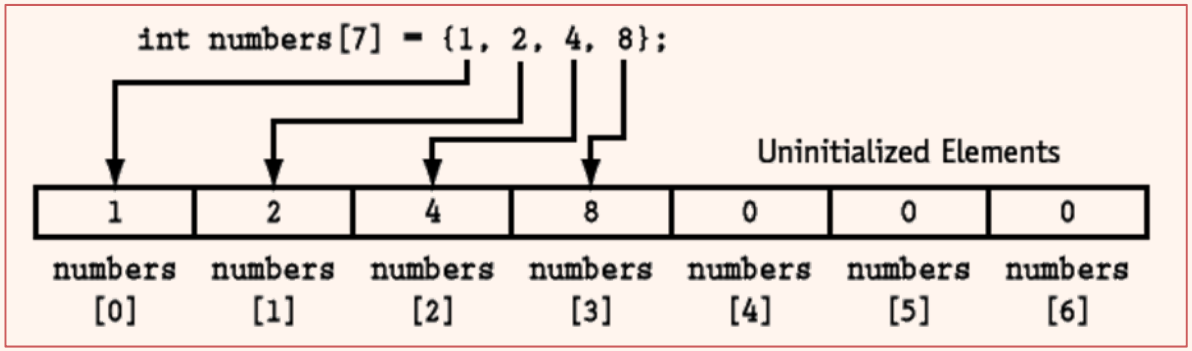
```
int main()
{ const int SIZE=5;                    // defines the size N for 5 elements
double a[SIZE];                        // declares the array's elements as type double
cout << "Enter " << SIZE << " numbers:\t";
for (int i=0; i<SIZE; i++)
cin >> a[i];
cout << "In reverse order: ";
for (int i=SIZE-1; i>=0; i--)
cout << "\t" << a[i];
cout<<endl;
}
```

```
Enter 5 numbers:        11 24 6 7 8
In reverse order:       8       7       6       24      11
```

روش‌های مقداردهی اولیه

- int a[6]={3,2,4,5,6,1};
- int number[7]={1,2,4,8};

باقی صفر می‌شوند.



- int a[6]={0};

تمامی خانه‌ها صفر می‌شوند.

- float a[ ]={22.2,44.4,66.6}

a

0	22.2
1	44.4
2	66.6

Implicit Array Sizing

در این روش کامپایلر با توجه به مقادیر داده شده بعد را در نظر می‌گیرد.

- float a[3] = { 22.2, 44.4 ,66.6 , 88.8 };

```
// ERROR: too many values!
```

برای تعریف یک ثابت دو روش کلی وجود دارد:

- استفاده از Const

```
Const int  n=15;
```

- استفاده از پیش‌پردازنده‌ی define

```
#define      n    15
```

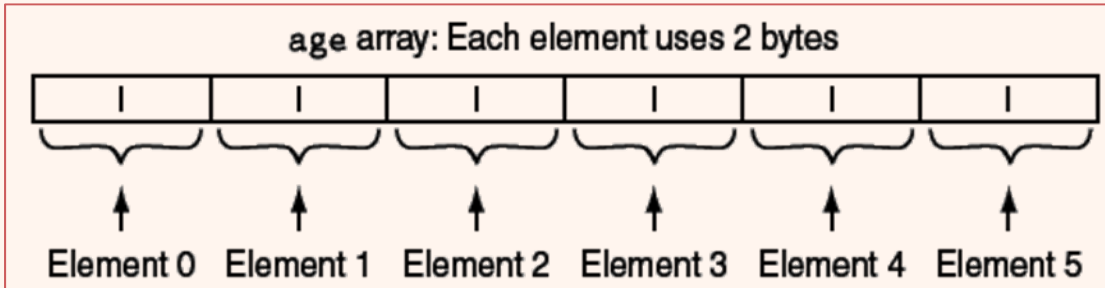
```
#include <iostream>
using namespace std;

int main(){
    const int arraySize = 10;
    int n[arraySize]={1,2,3,4,5,6,7,8,9,10}; // array s has 10 elements
    // initialize elements of array n to 0
    for ( int i = 0; i < arraySize; i++ )
        n[i]=0;
    cout << "Element\t\t\t" << "Value" << endl;
    // output each array element's value
    for ( int j = 0; j < arraySize; j++ )
        cout << j << "\t\t\t" << n[ j ] << endl;
    return 0;
}
```

```
Element                 Value
0                       0
1                       0
2                       0
3                       0
4                       0
5                       0
6                       0
7                       0
8                       0
9                       0
```

مثال

```
short age[6];
```

- اگر short دوبایت فضا نیاز داشته باشد:



ARRAY DECLARATION	NUMBER OF ELEMENTS	SIZE OF EACH ELEMENT	SIZE OF THE ARRAY
`char letter[26];`	26	1 byte	26 bytes
`short ring[100];`	100	2 bytes	200 bytes
`int mile[84];`	84	4 bytes	336 bytes
`float temp[12];`	12	4 bytes	48 bytes
`double distance[1000];`	1000	8 bytes	8,000 bytes

آرایه‌ی مقداردهی نشده

An Uninitialized Array

```
int main()
{ const int SIZE=4; // defines the size N for 4 elements
float a[SIZE]; // declares the array's elements as type float
for (int i=0; i<SIZE; i++)
      cout << "\ta[" << i << "] = " << a[i] << endl;
}
```

```
a[0] = -1.7606
a[1] = 3.98827e-34
a[2] = 1.1934e-38
a[3] = 1.19329e-38
```

اگر آرایه را مقداردهی نکنید ممکن است مقادیر زباله در حافظه وجود داشته باشد

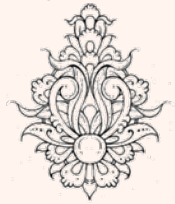
نکات

- نمی‌توان مقدار آن‌ها را به یکدیگر تخصیص داد:

```
float a[7] = { 7.2, 3.4, 4.6 };
float b[7] = { 53.3, 565.5, 7.7 };
b = a;    // ERROR
```

- نمی‌توانیم یک آرایه را به طور مستقیم برای مقداردهی اولیه به آرایه‌ی دیگر استفاده کنیم:

```
float a[7] = { 222.2, 344.4, 866.6 };
float b[7] = a;   // ERROR
```



خارج از محدوده

- در زبان C اگر آرایه‌ای را با ۴ عنصر تعریف نماییم، در این صورت هیچ تضمینی وجود ندارد که در برنامه به اندیس‌هایی فراتر از اندازه تعریف شده دست نیابد. (ممکن است به مقادیری دست یابد که بی‌ارزش هستند
- مثال:

```
#include <iostream>
using namespace std;int main()
{  const int SIZE=4; // defines the size N for 4 elements
   double a[SIZE]={ 33.3, 44.4, 55.5, 66.6 };// declares the array's elements as double

   for (int i=0; i<7; i++) //ERROR: index is out of bounds!
      cout << "\ta[" << i << "] = " << a[i] << endl;
}
```

```
a[0]  = 33.3
a[1]  = 44.4
a[2]  = 55.5
a[3]  = 66.6
a[4]  = -9.25596e+061
a[5]  = -9.25596e+061
a[6]  = 1.90727e-307
```

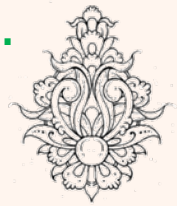
نکات

- به روی هر عنصر از آرایه همانند یک متغیر معمولی عملیات ریاضی می تواند صورت پذیرد.

```
Pay  =  hours [3] * rate ;
```

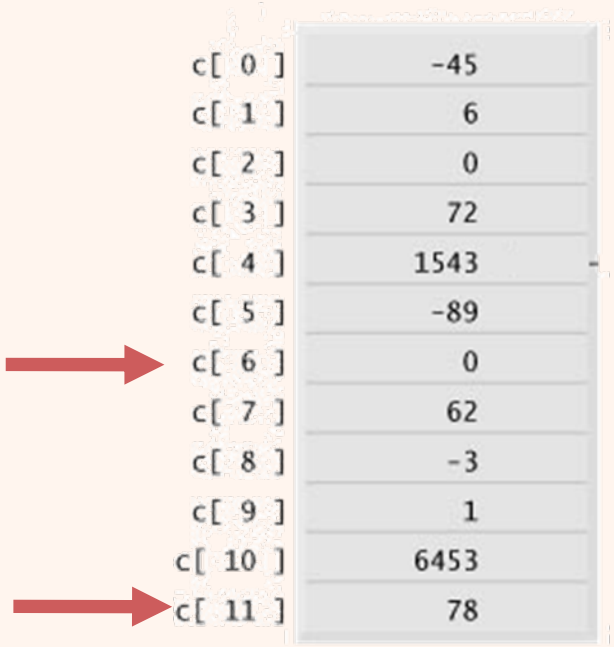
```
int score[5]  =  { 7 ,  8 ,  9 ,  10 ,  11} ;
++score[2] ;   // Pre-increment operation on the value in score [2]
score[4]++ ;   // Post-increment operation on the value in score [4]
```

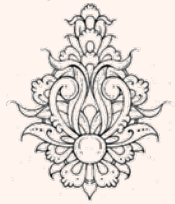
```
amount[count]-- ; //This decrements the value stored in amount [count].
amount[count--] ; //This decrements the variable count, but dose
                  //nothing to the value stored in amount [count].
```



- کروشه‌های دور اندیس آرایه یک عملگر در C++ محسوب می‌گردند.
- تقدم این کروشه‌ها با پرانتز برابر است.
- اگر a برابر با ۵ و b برابر با ۶ باشد مقدار عبارت زیر چند است؟

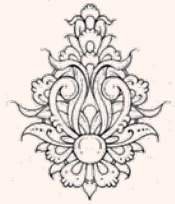
```
c[ a + b ] += 2;
x=c[6]/2+c[a+b];
```





```
int main()
 {
 const int NUM_WORKERS = 5;
 int worker;
 int hours[NUM_WORKERS];
 double payRate, grossPay;

 cout << "Enter the hours worked by " << NUM_WORKERS
 << " employees who all\n"
 << "earn the same hourly rate.\n";

 for (worker = 0; worker < NUM_WORKERS; worker++)
  {
     cout << "Employee #" << (worker+1) << ": ";
     cin >> hours[worker];
  }
 cout << "\nEnter the hourly pay rate for all the employees: ";
 cin >> payRate;

 cout << "\nHere is the gross pay for each employee:\n";

 for (worker = 0; worker < NUM_WORKERS; worker++)
  {
     double grossPay = hours[worker] * payRate;
     cout << "Employee #" << (worker+1);
     cout << ": $" << grossPay << endl;
  }
 return 0;
 }
```

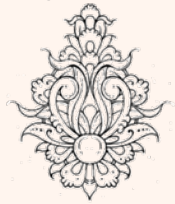
```
Enter the hours worked by 5 employees who all
earn the same hourly rate.
Employee #1: 12
Employee #2: 3
Employee #3: 44
Employee #4: 5
Employee #5: 6

Enter the hourly pay rate for all the employees: 12

Here is the gross pay for each employee:
Employee #1: $144
Employee #2: $36
Employee #3: $528
Employee #4: $60
Employee #5: $72
```



مثال

```
#include <iostream>
using namespace std;

int a[] = {1, 2, 3, 4, 40};
int n, result=0;

int main ()
{
  for ( n=0 ; n<5 ; n++ )
  {
    result += a[n];
  }
  cout << "The sum is:" <<result;
  return 0;
}
```

```
The sum is:50
```



تمرین

- اعدادی از یک آرایه را خوانده اطلاعات را به صورت هیستوگرام فراوانی رسم کنید.
  - برای نشان دادن هر عنصر از «*» استفاده نمایید.

{12,3,4,6,13,16,2,7,0,9}

```
Element Value   Histogram
0       12      ************
1       3       ***
2       4       ****
3       6       ******
4       13      *************
5       16      ****************
6       2       **
7       7       *******
8       0
9       9       *********
```