مبانی برنامه‌نویسی

(۱۳۹۹-۱۳-۱۱)

جلسه‌ی بیست‌ونهم

دانشگاه شهید بهشتی

پاییز ۱۳۹۲

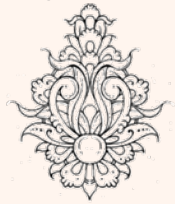
دانشکده‌ی مهندسی برق و کامپیوتر

احمد محمودی ازناوه

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

فهرست مطالب

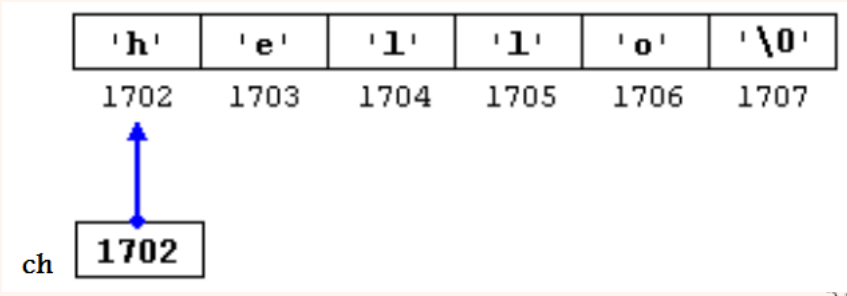
- اشاره‌گر
  - اشاره‌گر و آرایه
  - ارجاع و اشاره‌گر
  - انواع فراخوانی تابع



اشاره‌گرها و رشته‌ها

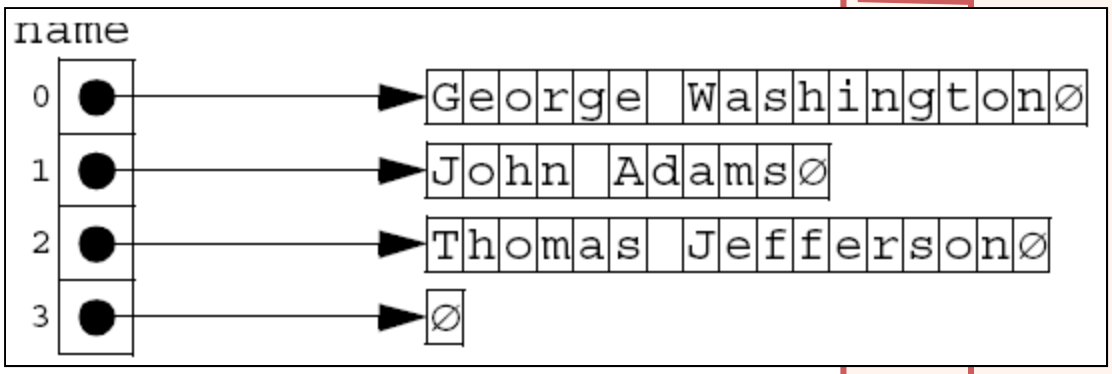
- رشته‌ها آرایه‌ای از جنس کاراکتر هستند.
- ارتباطی که میان آرایه و اشاره‌گرها وجود دارد برای رشته‌ها نیز صادق است.

char * ch=“hello”;



char * str[5]={“amir”, “maryam”, “ziba”, “ali”, “reza”};

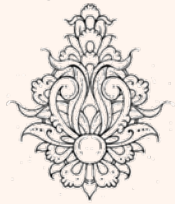
```cpp
#include <iostream>

using namespace std;
int main()
{
char buffer[80];
cin.getline(buffer,80,'$');
char* name[4];
name[0] = buffer;
int count = 0;
for (char* p=buffer; *p != '\0'; p++)
     if (*p == '\n')
        { *p = '\0'; // end name[count]
            name[++count] = p+1; // begin next name
         }
cout << "The names are:\n";
   for (int i=0; i<count; i++)
       cout << "\t" << i << ". [" << name[i] << "]" << endl;
}
```




```
George Washington
John Adams
Thomas Jefferson
$
The names are:
        0. [George Washington]
        1. [John Adams]
        2. [Thomas Jefferson]
```

انواع فراخوانی تابع

- در c++ به سه روش می‌توان آرگومان‌ها را به توابع ارسال نمود:
  - فراخوانی با **«مقدار»**
  - فراخوانی با ارجاع به وسیله‌ی **«آرگومان‌های ارجاعی»**
  - فراخوانی با ارجاع به وسیله‌ی **«اشاره‌گر»**



اشاره‌گر و ارجاع

Pointers

```
int i;
int *pi = &i;
```

References

```
int i;
int &ri = i;
```

```
*pi = 4;
```

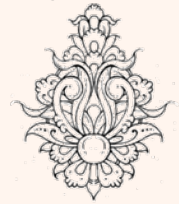
باید dereference شود

```
ri = 4;
```

به‌وسیله‌ی کامپایلر اتوماتیک dereference می‌گردد

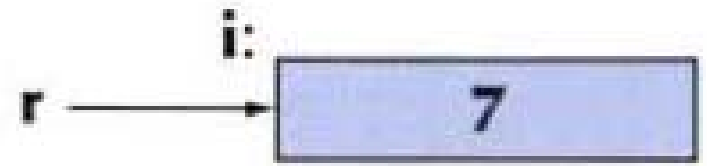
مرجع هم مانند اشاره‌گر می‌تواند حاوی آدرس یک متغیر باشد،

اشاره‌گر یک متغیر است و می‌تواند به متغیرهای متفاوتی اشاره کند، اما مرجع چنین نیست

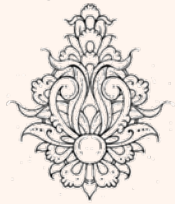


```
int main()
    {
int i = 7;
int&r = i;
r = 9;
i = 10;
cout << r <<' '<< i <<endl;
    }
```

```
10 10
```



- نام دیگری برای یک Object است.
  - زمان به وجود آمدن می‌باید مقداردهی گردد.
  - هنگامی که initialized گردد نمی‌تواند، مجدداً به Object دیگری اشاره کند.

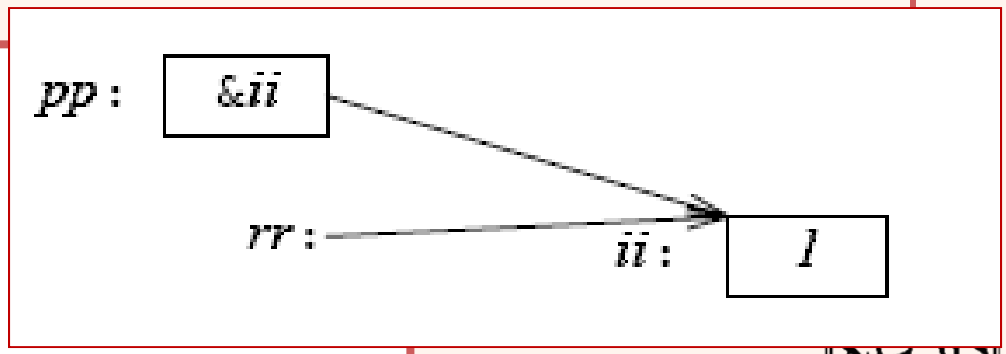


ارجاع (ادامه...)

```
int main()
{
     int i = 1 ;
     int &r = i ;
     int &r2 ;
     cout << r <<' '<< i <<endl;
}
```

```
: error C2530: 'r2' : references must be initialized
```

```
int main()
{
     int ii = 0 ;
     int & rr = ii ;
     rr ++;
     int * pp = &rr ;
     *pp = 12;
     cout << rr <<' '<< ii << ' ' << *pp << endl;
}
```




```
12 12 12
Press any ke
```

ارجاع ثابت

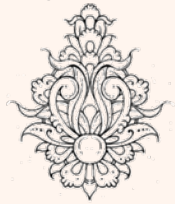
Const Reference را از طریق Objectیک نمی‌توان تغییر داد.

```
int main()
    {
int i = 7;
int& r = i;
r = 9;
const int& cr = i;
cr = 7;
i = 11;
cout << cr << endl;
cout << r << endl;
cout << i << endl;
    }
```

```
: error C3892: 'cr' : you cannot assign to a variable that is const
```

```
int main()
    {
int i = 7;
int& r = i;
r = 9;
const int& cr = i;
i = 11;
cout << cr << endl;
cout << r << endl;
cout << i << endl;
    }
```

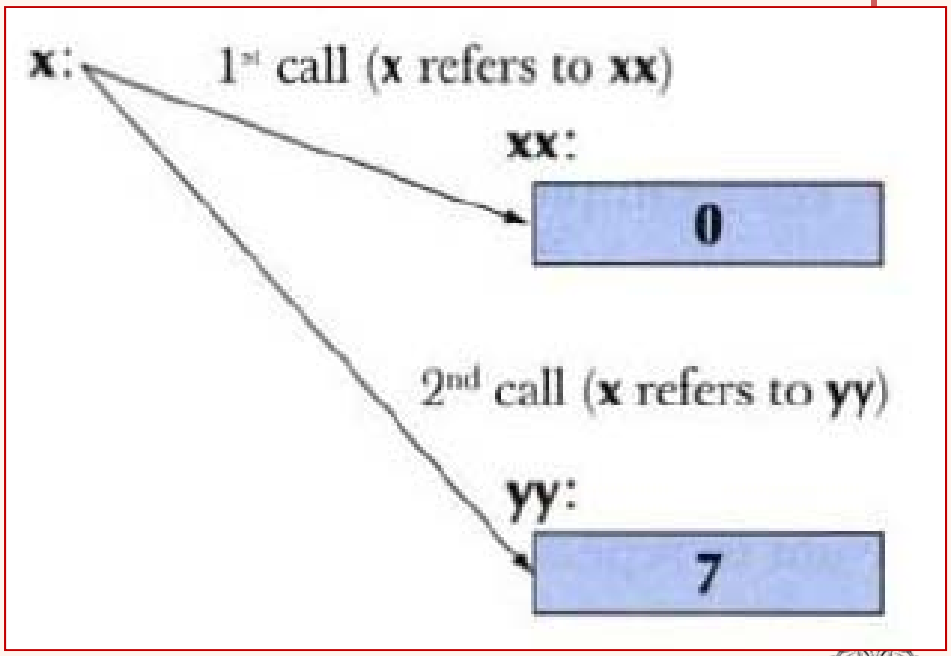
```
11
11
11
```



فراخوانی با ارجاع

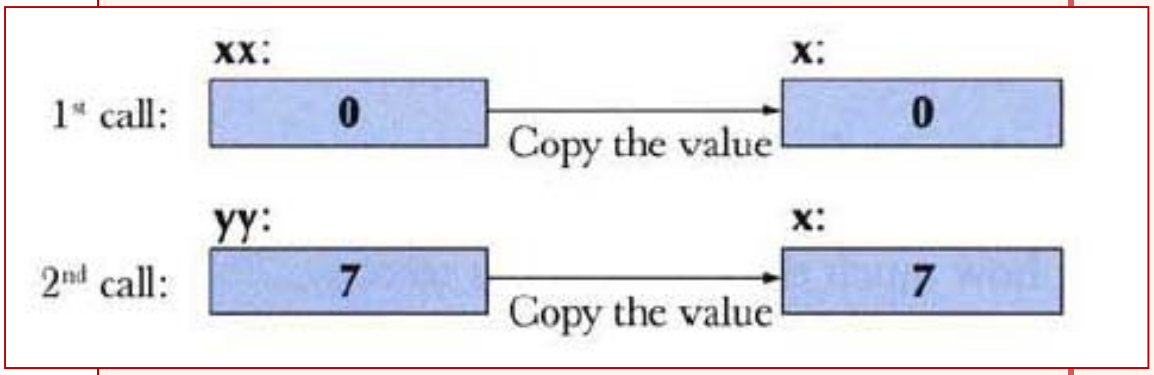
```
int f(int& x)
{
  x = x+1;
  return x;
}
int main()
    {
int xx=0;
cout << f(xx)<<endl;
cout << xx << endl;
int yy = 7;
cout << f(yy)<< endl;
cout << yy << endl;
}
```

```
1
1
8
8
```

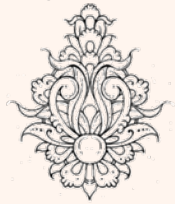


فراخوانی با مقدار

```
int f(int x)
{
   x = x+1;
   return x;
}
int main()
     {
int xx=0;
cout << f(xx)<<endl;
cout << xx << endl;
int yy = 7;
cout << f(yy)<< endl;
cout << yy << endl;
}
```




```
1
0
8
7
```



مثال

```
void f(int a, int& r, const int& cr){
   ++a; ++r; ++cr;
}
void g(int a, int& r, const int& cr){
   ++a; ++r; int x = cr; ++x;
 }

int main(){
   int x = 0;
   int y = 0;
   int z = 0;
   g(x,y,z);
   g(1,2,3);
   g(1,y,3);
}
```

error: cr is const

x==0; y==1; z==0

error: reference argument r needs a variable to refer to

ok: since cr is const we can pass “a temporary”

فراخوانی به وسیله‌ی اشاره‌گر

```
void swapnum(int&,int&);
int main() {
  int a = 10;
  int b = 20;

  swapnum(a,b);
  cout<<" a is:" << a << "\t b is:" <<b <<endl;
  return 0;
}
void swapnum(int& i, int& j)
  int temp;
  temp = i;
  i = j;
  j = temp;
}
```

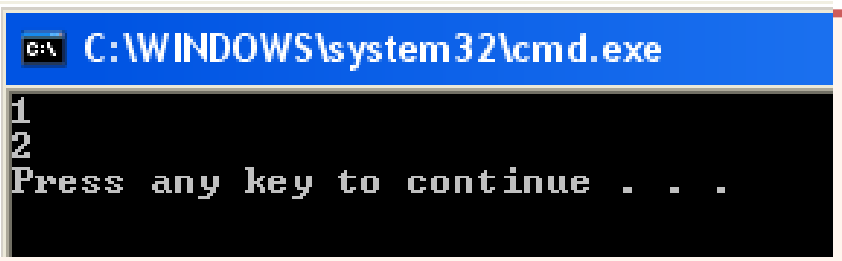
```
a is:20         b is:10
```

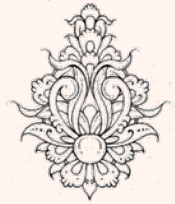
```
void swapnum(int*,int*);
int main() {
  int a = 10;
  int b = 20;

  swapnum(&a,&b);
  cout<<" a is:" << a << "\t b is:" <<b <<endl
  return 0;
}
void swapnum(int *i, int *j) {
  int temp;
  temp = *i;
  *i = *j;
  *j = temp;
}
```

```
a is:20         b is:10
```

مثال

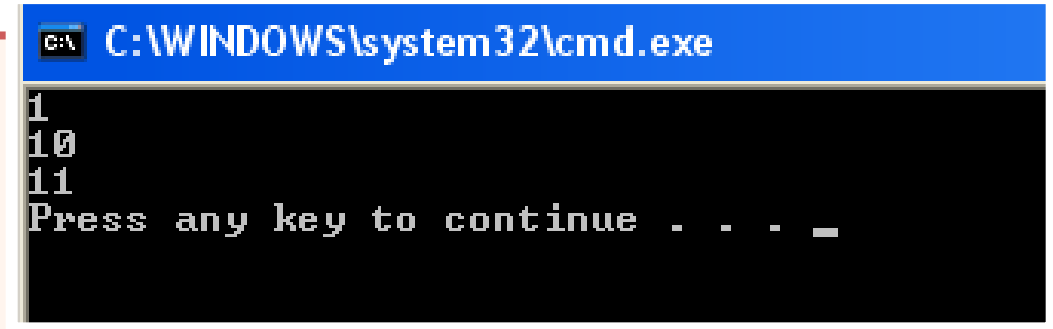
```
#include <iostream>
using namespace std;
int* f(int* x) {
     (*x)++;
     return x;
}
int& g(int& x) {
     x++; // Same effect as in f()
     return x;
}
int main() {
     int a = 0;
     cout<<*f(&a)<<endl; // Ugly (but explicit)
     cout<<g(a)<<endl; // Clean (but hidden)
}
```

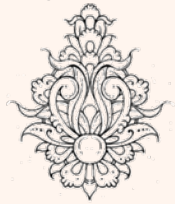




مثال

```
int* f(int* x) {
    (*x)++;
    return x;
}
int& g(int& x) {
    x++; // Same effect as in f()
    return x;
}
int main() {
    int a = 0;
    cout<<*f(&a)<<endl; // Ugly (but explicit)
    cout<<(g(a)=10)<<endl; // Clean (but hidden)
    g(a);
    cout<<a<<endl;
}
```





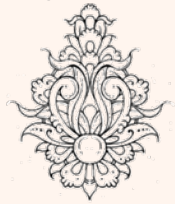
تمرین

تابعی بنویسید که رشته‌ای را به عنوان ورودی گرفته (یک اشاره‌گر به کاراکتر) طول آن را بازگرداند.

```
int strln(char *s)
```

```
using namespace std;
int strln(char *);
int main ()
{
  char a[50];
  cout << "Enter string:";
  cin.getline(a,50);
  cout << "The length of the string is:"<<strln(a)<<endl;
    }
int strln(char *s)
    {
    char *p=s;
    while(*p!='\0')
        p++;
    return p-s;
    }
```

```
Enter string:This is a test
The length of the string is:14
```



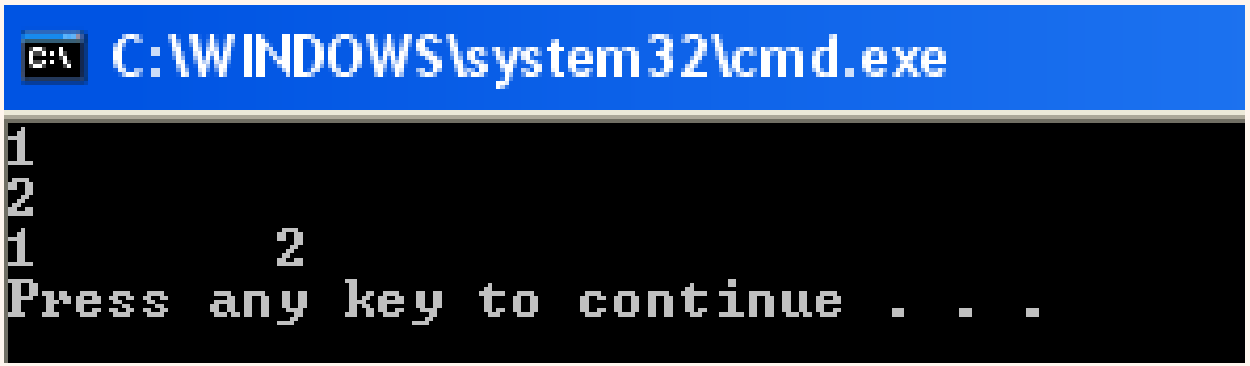
مثال

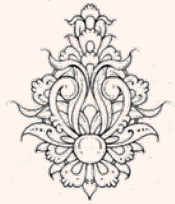
```
#include <iostream>
 using namespace std;


int main(void) {
    int n = 108;                    // an int
    int* p = &n;                   // a pointer to an int
    cout<< "The address of p is:"<< p <<"and the value is:"<<*p <<endl;
    ++(*p);                        // OK: increments int *p
    cout<< "The value of *p is:"<<*p <<endl;
    ++p;                            // OK: increments of address p
    cout<< "The address of p is:"<< p <<"and the value is:"<<*p <<endl;
        }//end main
```

```
The address of p is:0012FF28and the value is:108
The value of *p is:109
The address of p is:0012FF2Cand the value is:-858993460
```

مثال

```
int main()
{
    int n[]={1,2,3,4};
    int *p;
    p=n;
    cout << *p++ <<endl;
    cout << *p <<endl;
    cout<< n[0]<<'\t'<<n[1]<<endl;
}
```

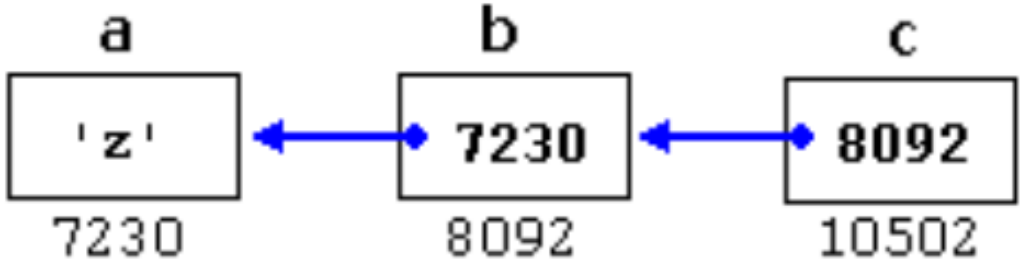




اشاره‌گر به اشاره‌گر

- اگر اشاره‌گری آدرس اشاره‌گر دیگری را در خود ذخیره نماید، به آن «اشاره‌گر به اشاره‌گر» می‌گویند.
- ساختار تعریف اشاره‌گر به اشاره‌گر به صورت زیر است:

<نوع> ** نام متغیر;



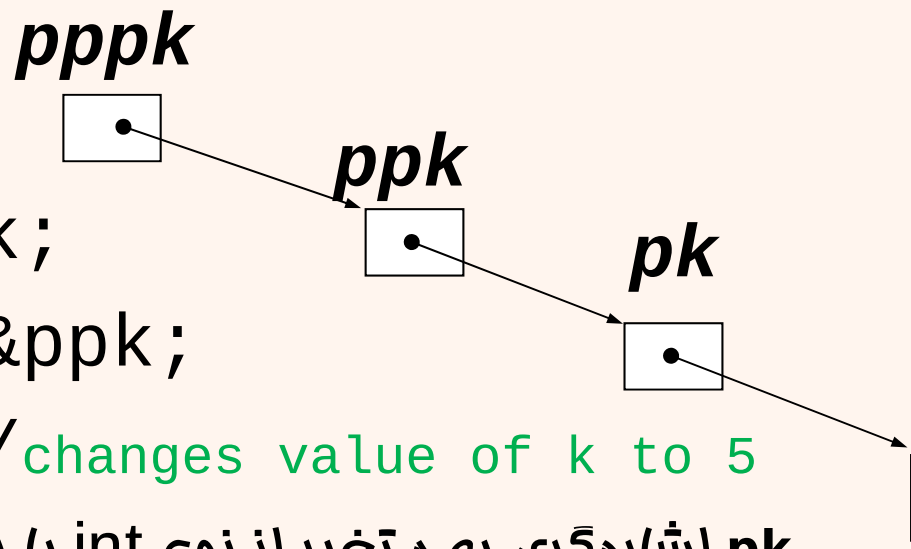

```
char a;
char * b;
char ** c;
a = 'z';
b = &a;
c = &b;
```

- c has type char** and a value of 8092
- *c has type char* and a value of 7230
- **c has type char and a value of 'z'

اشاره‌گر به اشاره‌گر

- **یک اشاره‌گر می‌تواند به اشاره‌گر دیگری اشاره کند.**
- **مثال:**

```
int k = 3;
int* pk = &k;
int** ppk = &pk;
int*** pppk = &ppk;
***pppk = 5; //changes value of k to 5
```



- **pk** اشاره‌گری به متغیر از نوع int با نام k است.
- ppk اشاره‌گری به اشاره‌گر pk است و اشاره‌گر pppk هم به اشاره‌گر ppk اشاره دارد.

مثال

- نتیجه‌ی کد زیر چیست؟

```
#include <iostream>
  using namespace std;
int main()
    {
      float x,y;
      int *p;
      x=3.1415;
      p = &x;
      y = *p;

    cout << "The address of x is " << p<<endl;
    cout<< "\nThe value of x is " << y <<endl;

      return 0; // indicates successful termination
   } // end main*/
```

```
1>e:\ac++\code\pointers\pointers\5th.cpp(7) : warning C4305: '=' : truncation from 'double' to 'float'
1>e:\ac++\code\pointers\pointers\5th.cpp(8) : error C2440: '=' : cannot convert from 'float *__w64 ' to 'int *'
1>        Types pointed to are unrelated; conversion requires reinterpret_cast, C-style cast or function-style cas
1>e:\ac++\code\pointers\pointers\5th.cpp(9) : warning C4244: '=' : conversion from 'int' to 'float', possible loss
```