



Fundamentals of Object Oriented Programming

CSN- 103

Dr. R. Balasubramanian

Associate Professor

Department of Computer Science and Engineering

Indian Institute of Technology Roorkee

Roorkee 247 667

balarfcs@iitr.ac.in

<https://sites.google.com/site/balaiiitr/>





Arrays

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7      int a[20];
8      a[0]=1;
9      a[1]=7;
10     a[2]=2;
11     a[3]=9;
12
13     for (int i=0; i<4; i++)
14     {
15         cout<<" "<<a[i];
16     }
17
18     for (int i=4; i<20; i++)
19     {
20         a[i]=a[i-4]+a[i-3];
21         cout<<" "<<a[i];
22         if (i==19)
23             cout<<endl;
24     }
25
26     return 0;
27 }
```

Terminal

```
sh-4.3$ g++ HRseries.cpp  
sh-4.3$ a.out  
 1 7 2 9 8 9 11 17 17 20 28 34 37 48 62 71 85 110 133 156  
sh-4.3$
```



References Operator (&) in C++

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7      int sum, a, b;
8      cout<<"enter a"<<endl;
9      cin>>a;
10     cout<<"enter b"<<endl;
11     cin>>b;
12
13     sum=a+b;
14     cout<<"sum of two numbers="<<sum<<endl;
15     cout<<"Address of a is "<<&a<<endl;
16     cout<<"Address of b is "<<&b<<endl;
17     cout<<"Address of sum is "<<&sum<<endl;
18
19     return 0;
20 }
```

Terminal

```
sh-4.3$ g++ add2num.cpp
sh-4.3$ a.out
enter a
20
enter b
40
sum of two numbers=60
Address of a is 0x7fffc9af9b88
Address of b is 0x7fffc9af9b84
Address of sum is 0x7fffc9af9b8c
sh-4.3$
```

Reference Operator in C++


```
void main()  
{  
    int n=44;  
    int rn;  
    rn=n;  
    cout<<n<<rn<<endl;  
    n--;  
    cout<<n<<rn<<endl;  
    rn*=2;  
    cout<<n<<rn<<endl;  
}
```

```
void main()  
{  
    int n=44;  
    int& rn=n;  
    cout<<n<<rn<<endl;  
    n--;  
    cout<<n<<rn<<endl;  
    rn*=2;  
    cout<<n<<rn<<endl;  
}
```

Dereference Operator (*) in C++

A pointer is a variable that points to an address of an another variable.

```
void main()  
{  
    int u=30;  
    int v;  
    int *pu, *pv;  
    pu=&u;  
    v=*pu;  
    pv=&v;  
    cout<<u<<&u<<pu<<*pu;  
    cout<<v<<&v<<pv<<*pv;  
}
```

A faint, stylized illustration of a lion lying down, facing left, positioned behind the C++ code block.

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7      int u=30;
8      int v;
9      int *pu, *pv;
10     pu=&u;
11     v=*pu;
12     pv=&v;
13     cout<<" " <<u<<" " <<&u<<" " <<pu<<" " <<*pu<<endl;
14     cout<<" " <<v<<" " <<&v<<" " <<pv<<" " <<*pv<<endl;
15
16     return 0;
17 }
```

Terminal

```
sh-4.3$ g++ pointcheck0.cpp
sh-4.3$ a.out
30 0x7fffa3a4ec8c 0x7fffa3a4ec8c 30
30 0x7fffa3a4ec88 0x7fffa3a4ec88 30
sh-4.3$
```

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7      int v=3;
8      int* pv;
9      pv=&v;
10     cout<<" "<<*pv<<" "<<v<<endl;
11     *pv=5;
12     cout<<" "<<*pv<<" "<<v<<endl;
13     return 0;
14 }
```

Terminal

```
sh-4.3$ g++ pointcheck2.cpp
sh-4.3$ a.out
 3 3
 5 5
sh-4.3$
```


Null Pointer in C++

```
void main()  
{  
    int *pv;  
    pv=0; //pv=NULL;  
    cout<<*pv;  
}
```



```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7      int* pv;
8      pv=NULL;
9      cout<<*pv;
10
11     return 0;
12 }
13
```

Terminal

```
sh-4.3$ g++ pointcheck1.cpp
sh-4.3$ a.out
Segmentation fault (core dumped)
sh-4.3$
```



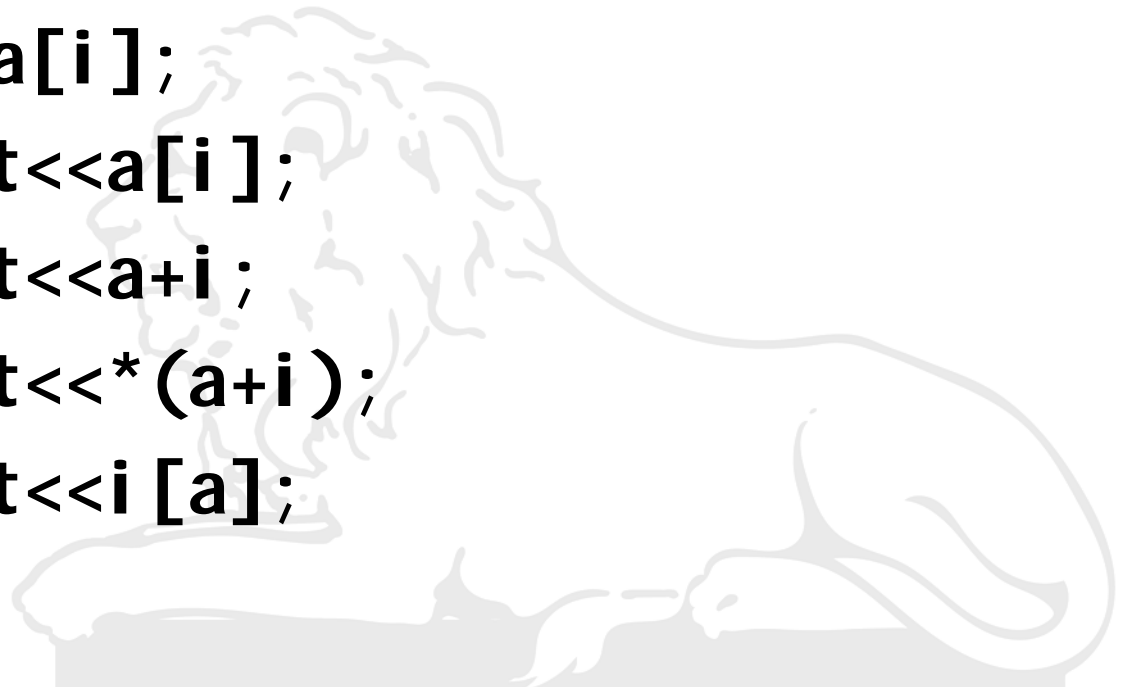
Dangling Pointer in C++

```
void main()  
{  
  int u1,u2;  
  int v=3;  
  int *pv;  
  u1=2*(v+5);  
  u2=2*( *pv+5);  
  cout<<u1<<u2;  
}
```



Arrays and Pointers in C++

```
int a[10];  
for (int i=0; i<10; i++)  
    {cin>>a[i];  
      cout<<a[i];  
      cout<<a+i;  
      cout<<*(a+i);  
      cout<<i[a];  
    }
```

A faint, light-gray background image of a lion statue, likely the Rooster Lion of IIT Roorkee, is visible behind the code.
