

Theory (Constructor)

Constructor function is a function of C++ which is to be included in a class declaration. A class's constructor is called each time an object of that class is created. Thus any initializations that need to be performed on an object can be done automatically by the constructor function. The name of the constructor function of a program is same as the name of that program's class name. It has no return type and it is a part of a class. When writing a program for very long and many parts are included in that program, constructor function is needed to initialize the objects.

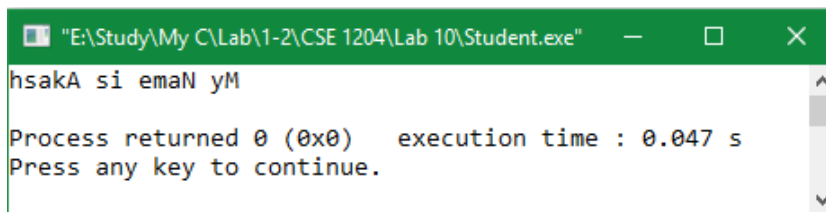
Code

```
#include<bits/stdc++.h>
using namespace std;

class StrRev {
    string name;
public:
    StrRev(string str) {
        name = str;
        reverse(name.begin(),name.end());
    }
    show(){
        cout<<name<<endl;
    }
};

int main()
{
    StrRev ob("My Name is Akash");
    ob.show();
    return 0;
}
```

Output



```
"E:\Study\My C\Lab\1-2\CSE 1204\Lab 10\Student.exe"
hsakA si eMaN yM
Process returned 0 (0x0)   execution time : 0.047 s
Press any key to continue.
```

Comment

Here a string was initialized by the object of class StrRev. And then using a constructor the string was reversed and finally it was shown in the output.

Theory (Copy Constructor)

When a copy of an argument is made during a function call, the normal constructor is *not* called. Instead, the object's **copy constructor** is called. A copy constructor defines how a copy of an object is made. It can be explicitly defined for a class that you create . However, if a class does not explicitly define a copy constructor, as is the case here, then C++ provides one by default. The default copy constructor creates a bitwise (that is, identical) copy of the object.

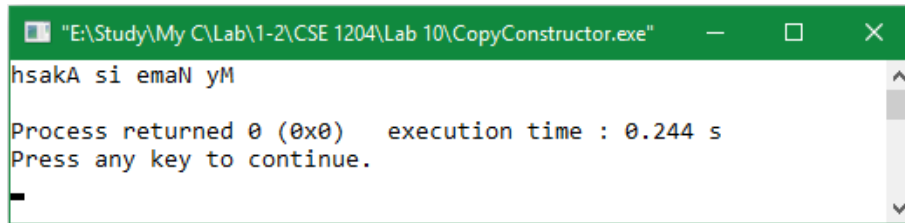
Code

```
#include<bits/stdc++.h>
using namespace std;

class StrRev {
    string name;
public:
    StrRev(string str)
    {
        name = str;
    }
    StrRev(const StrRev &ob)
    {
        name = ob.name;
        reverse(name.begin(),name.end());
    }
    show()
    {
        cout<<name<<endl;
    }
};

int main()
{
    StrRev ob("My Name is Akash");
    StrRev ob2=ob;
    ob2.show();
    return 0;
}
```

Output



```
"E:\Study\My C\Lab\1-2\CSE 1204\Lab 10\CopyConstructor.exe"
hsakA si emaN yM
Process returned 0 (0x0) execution time : 0.244 s
Press any key to continue.
```

Comment

Here two constructors were declared and the second one is the copy constructor as it has the address of an object in its parameter. And in the main() function, another object was taken and the first object was assigned to the second object.

Theory (Inheritance)

Inheritance is the process by which one object can acquire the properties of another specifically, an object can inherit a general set of properties to which it can add those features that are specific only to itself. Inheritance is important because it allows an object to support the concept of *hierarchical classification*. So, Inheritance certainly plays a very important role in OOP.

Code

```
#include<bits/stdc++.h>
using namespace std;
class base{
public:
    string n;
    int s;
    get(){
        cout<<"Enter name:";
        cin>>n;
        cout<<"Enter salary:";
        cin>>s;
    }
};
class derive : public base{
public:
    string name;
    int salary;
    get_sn(base ob)
    {
```

```

        name = ob.n;
        salary = ob.s;
    }
    show(){
        cout<<"Name:"<<name<<"\nsalary:"<<salary<<endl;
    }
};
int main()
{
    base b;
    derive d;
    b.get();
    d.get_sn(b);
    d.show();
    return 0;
}

```

Output

```

E:\Study\My C\Lab\1-2\CSE 1204\Lab 10\Inheritance.exe
Enter name: Akash
Enter salary: 40000
Name: Akash
salary: 40000

Process returned 0 (0x0)   execution time : 12.666 s
Press any key to continue.

```

Comment

Here, a base class was taken where name and salary of a person was taken from the user by get() method. And after that, a derive class was taken and the base class was inherited there. Also in the derive class there was declared a function named get_sn() and the base class object was used as the parameter there. And finally in the main() function, a base class and a derive class object was declared and the functions were called to show the required output.