# TASK 6:

**SHALLOW COPY:**

#include <iostream>

#include <string>

using namespace std;

class Person

{

string name;

int age;

public:

Person(){} //Default Constructor

Person(string name, int age) //Parametrized Constructor

{

this->name = name;

this->age = age;

}

void print() //Display Function

{

cout << "Name: " << name << endl;

cout << "Age: " << age << endl;

}

};

int main()

{

//Shallow Copy

Person p1("Huzaifa", 20); //Creating an object

Person p2 = p1; // Creating another object and copying previous one

p1.print();

p2.print(); // Object is copied through shallow copy and both will have same values

# }

# OUTPUT:

**DEEP COPY:**

#include <iostream>

#include <string>

using namespace std;

class Person

{

string name;

int \*age;

public:

Person() //Default Constructor

{

age = new int;

}

void setter(string nam, int ag) //Setter Function

{

name = nam;

\*age = ag;

}

void print() //Display Function

{

cout << "Name: " << name << endl;

cout << "Age: " << \*age << endl;

}

Person(Person& p) //Parametrized Constructor for Deep Copy

{

name = p.name;

age = new int;

\*age = \*(p.age);

}

~Person() //Destructor

{

delete age;

}

};

int main()

{

//Deep Copy

Person p1; //Creating an object

p1.setter("Huzaifa", 20);

Person p2 (p1); // Creating another object and copying previous one to the new object

p1.print();

p2.print(); // Object is copied through deep copy and both will have same values

}

# OUTPUT:

