National University of Sciences and Technology (NUST)
School of Electrical Engineering and Computer Science

Department Of Electrical Engineering and Computer Sciences

Instructor: Mehreen Tahir **Date:**

Lab Engineer: Mehwish Kiran **Time:** 10:00am – 12:50pm

CS 212: Object Oriented Programming

Lab 06: Constructors

Information	Description
Name:	Irfa Farooq
CMS ID:	412564
Class:	BEE-14
Section:	D
Tenure:	Fall 2023

Task 1: Write a student class in C++
A student object should hold information about a student in your grade book and should have the following details:

- **string name**
- **int age**
- **int *p**

Code:

Student.h:

```
#include <iostream>

class Student {
public:
	Student();
	Student(Student& s1);
	void set_name(std::string);
	std::string get_name();
	void set_age(int);
	int get_age();
	void set_marks();
	void get_marks();
	void Print_Functions();

private:
	std::string name;
	int age;
	int* p;
};
```

Student_Information.cpp:

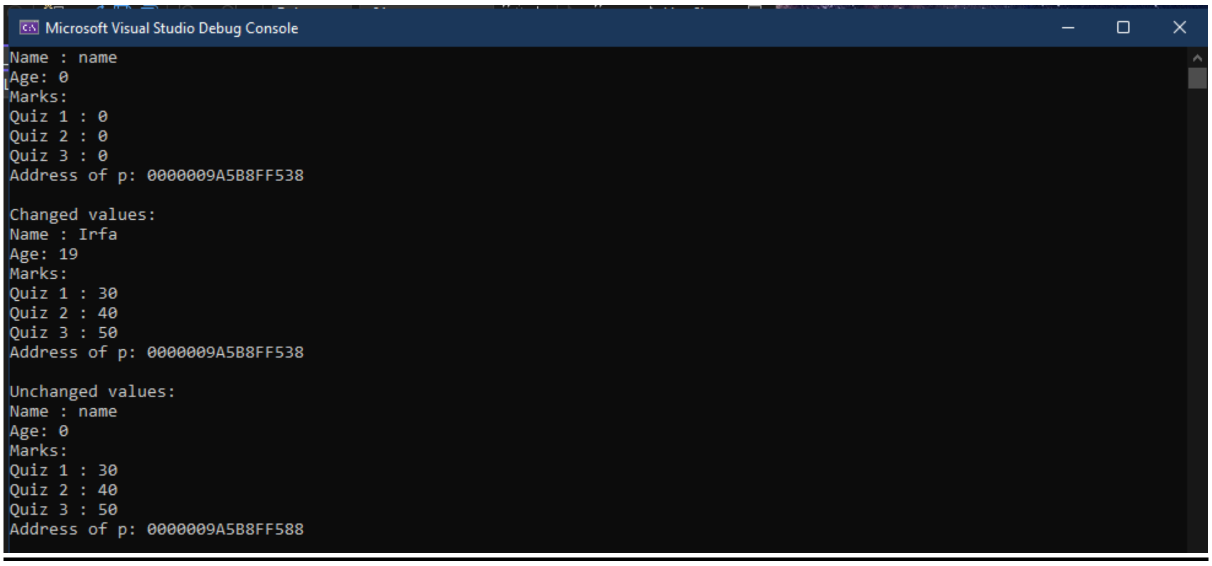
```
#include <iostream>
#include "Student.h"

Student::Student() {      //Default constructor for initializing values
	name = "name";
	age = 0;
	p = new int[3];
	*p = 0;
	*(p + 1) = 0;
	*(p + 2) = 0;
}

Student::Student(Student& s1) {
	name = s1.name;
	age = s1.age;
	p = new int[3];
	*p = *(s1.p);
	*(p + 1) = *((s1.p) + 1);
	*(p + 2) = *((s1.p) + 2);
}
```

```
void Student::set_name(std::string n){
	name = n;
}

std::string Student::get_name() {
	return name;
}

void Student::set_age(int a) {
	age = a;
}

int Student::get_age() {
	return age;
}

void Student::set_marks() {
	*p = 30;
	*(p + 1) = 40;
	*(p + 2) = 50;
}

void Student::get_marks() {
	for (int i = 0; i < 3; i++) {
		std::cout << "Quiz " << i + 1 << " : " << * (p + i) << std::endl;
	}
}

void Student::Print_Functions() {
	std::cout << "Name : " << get_name() << std::endl;
	std::cout << "Age: " << get_age() << std::endl;
	std::cout << "Marks: " << std::endl;
	get_marks();
	std::cout << "Address of p: " << &p << std::endl;
	std::cout << std::endl;
 }
```

Student_Main:

```
#include <iostream>
#include "Student.h"

int main() {
	Student s1;
	//Student s2 = s1;   //Default copy constructor
	Student s2(s1);      //User defined copy constructor
	std::cout << "Default values: " << std::endl;
	s1.Print_Functions();
	std::cout << "Changed values: " << std::endl;
	s1.set_name("Irfa");
	s1.set_age(19);
	s1.set_marks();
	s1.Print_Functions();
	std::cout << "Unchanged values: " << std::endl;
	s2.Print_Functions();
	return 0;
}
```

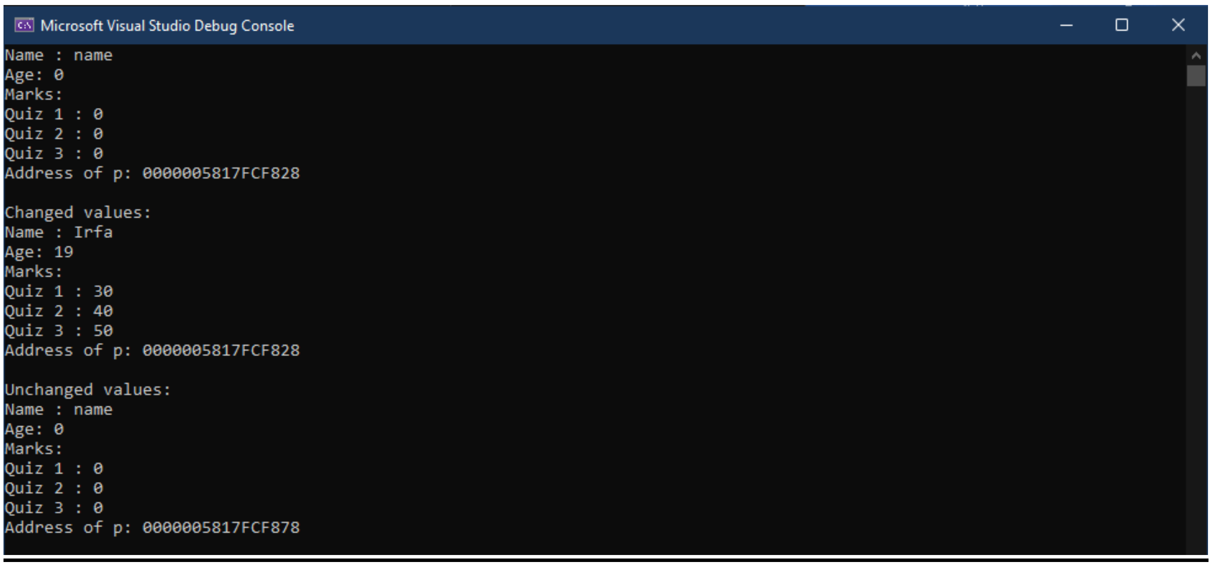
<u>Output Screenshots</u>



Microsoft Visual Studio Debug Console

```
Name : name
Age: 0
Marks:
Quiz 1 : 0
Quiz 2 : 0
Quiz 3 : 0
Address of p: 0000009A5B8FF538

Changed values:
Name : Irfa
Age: 19
Marks:
Quiz 1 : 30
Quiz 2 : 40
Quiz 3 : 50
Address of p: 0000009A5B8FF538

Unchanged values:
Name : name
Age: 0
Marks:
Quiz 1 : 30
Quiz 2 : 40
Quiz 3 : 50
Address of p: 0000009A5B8FF588
```

Figure 1: Default Copy Constructor Output



Microsoft Visual Studio Debug Console

```
Name : name
Age: 0
Marks:
Quiz 1 : 0
Quiz 2 : 0
Quiz 3 : 0
Address of p: 0000005817FCF828

Changed values:
Name : Irfa
Age: 19
Marks:
Quiz 1 : 30
Quiz 2 : 40
Quiz 3 : 50
Address of p: 0000005817FCF828

Unchanged values:
Name : name
Age: 0
Marks:
Quiz 1 : 0
Quiz 2 : 0
Quiz 3 : 0
Address of p: 0000005817FCF878
```

Figure 2: User-defined Copy Constructor Output

<u>Conclusion:</u>

In this lab, we were able to observe the differences between user-defined copy constructors and default copy constructors. The user-defined copy constructors allow a deep copy of the class objects and stores them accordingly in the copied objects while the default copy constructors allow a shallow copy of the class objects and stores only the addresses of the objects and attributes. To conclude, we were able to learn the usefulness of both kinds of constructors and their drawbacks.