DS Lab 01

24K-0923 Aqsa

Q1:

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

class BankAccount{
	private:
		double balance;
	public:
		//1. default Constructor
		BankAccount(){
			balance=0.0;
		}
		//2. Parameterized Constructor
		BankAccount(double initial_balance){
			balance= initial_balance;
		}
		//3. Copy Constructor
		BankAccount(const BankAccount &other){
			balance=other.balance;
		}
		// Function to withdraw money
```

```
        void withdraw(double amount){
            if(amount<=balance){
                balance -= amount;
            }else{
                cout << "Insufficient balance!" << endl;
            }
        }
        // Function to display balance
        void displayBalance() const {
        cout << "Balance: $" << balance << endl;
        }
};
int main(){
    // Default constructor called
    BankAccount acc1;
    cout << "Account_1: ";
    acc1.displayBalance();

    // Parameterized Constructor called
    BankAccount acc2(2000);
    cout << "Account_2: ";
    acc2.displayBalance();

    //Copy Constructor called
```

```
    BankAccount acc3(acc2);
    acc3.withdraw(500);
    cout <<"Account3 after withdrawing $500: ";
    acc3.displayBalance();
    return 0;
}
```

Output:

```
C:\Users\aqsas\OneDrive\Des

Account_1: Balance: $0
Account_2: Balance: $2000
Account3 after withdrawing $500: Balance: $1500

--------------------------------
Process exited after 0.1803 seconds with return value 0
Press any key to continue . . .
```

Q2:

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

class Exam {
private:
    char* student_Name;
    char* exam_Date;
    int score;

public:
    // Constructor
    Exam(const char* name, const char* date, int s) {
        student_Name = new char[strlen(name) + 1];
        strcpy(student_Name, name);

        exam_Date = new char[strlen(date) + 1];
        strcpy(exam_Date, date);

        score = s;
    }
```

```
    // No Copy Constructor (default shallow copy will be
used)
    // No Assignment Operator (default shallow copy will be
used)

    // Destructor
    ~Exam() {
        delete[] student_Name;
        delete[] exam_Date;
        cout << "Destructor called for Exam object!" << endl;
    }

    void display() const {
        cout << "Student: " << student_Name
            << ", Date: " << exam_Date
            << ", Score: " << score << endl;
    }
};

int main() {
    Exam exam1("Aqsa", "20-08-2025", 90);
    cout << "Exam1 details: ";
    exam1.display();
```

```
    // Shallow Copy happens here (default copy constructor)
    Exam exam2 = exam1;
    cout << "Exam2 (shallow copy of Exam1) details: ";
    exam2.display();

    // When program ends, both exam1 and exam2 destructors run
    // Problem: both try to delete the same memory
    return 0;
}
```

Output:

```
C:\Users\aqsas\OneDrive\Des

Exam1 details: Student: Aqsa, Date: 20-08-2025, Score: 90
Exam2 (shallow copy of Exam1) details: Student: Aqsa, Date: 20-08-2025, Score: 90
Destructor called for Exam object!

--------------------------------
Process exited after 2.737 seconds with return value 3221226356
Press any key to continue . . .
```

Q3:

```
#include <iostream>
#include <cstring>

using namespace std;

class Box {
private:
    int* data; // Pointer to an integer

public:
    // Constructor
    Box(int value) {
        data = new int(value);
        cout << "Constructed Box with value: " << *data <<
endl;
    }

    // Destructor
    ~Box() {
        delete data;
        cout << "Destructed Box" << endl;
```

```
    }

    // Copy Constructor (Deep Copy)
    Box(const Box& other) {
        data = new int(*other.data);
        cout << "Copy constructed Box with value: " << *data
<< endl;
    }

    // Copy Assignment Operator (Deep Copy)
    Box& operator=(const Box& other) {
        if (this != &other) { // Check for self-assignment
            delete data; // Free existing resource
            data = new int(*other.data); // Allocate new memory
and copy the value
            cout << "Copy assigned Box with value: " << *data
<< endl;
        }
        return *this;
    }

    // Function to get the value
    int getValue() const {
        return *data;
    }
```

```
};

void demonstrateShallowCopy() {
    cout << "Demonstrating shallow copy:" << endl;
    Box box1(90);
    Box box2 = box1; // This will invoke the copy constructor
(Deep Copy)

    cout << "Box1 value: " << box1.getValue() << endl;
    cout << "Box2 value: " << box2.getValue() << endl;
}

void demonstrateDeepCopy() {
    cout << "Demonstrating deep copy:" << endl;
    Box box1(20);
    Box box2(30);

    box2 = box1; // This will invoke the copy assignment
operator (Deep Copy)

    cout << "Box1 value: " << box1.getValue() << endl;
    cout << "Box2 value: " << box2.getValue() << endl;
}
```

```
int main() {
    demonstrateShallowCopy();
    demonstrateDeepCopy();
    return 0;
}
```

Output:

```
C:\Users\aqsas\OneDrive\Des

Demonstrating shallow copy:
Constructed Box with value: 90
Copy constructed Box with value: 90
Box1 value: 90
Box2 value: 90
Destructed Box
Destructed Box
Demonstrating deep copy:
Constructed Box with value: 20
Constructed Box with value: 30
Copy assigned Box with value: 20
Box1 value: 20
Box2 value: 20
Destructed Box
Destructed Box

--------------------------------
Process exited after 0.3055 seconds with return value 0
Press any key to continue . . .
```