MCA Semester-II
Subject: Advanced Programming Technique-I [C++]

Practical Laboratory - 4

Topic/Unit: Constructor, Destructor, Copy constructor, Default constructor

Sr.	Write?	Task

1	*	In the `Student` class, add following constructors:
		`Student(); //default constructor`
		`Student(int rno, char nm[], float cmarks, float cppmarks, float javamarks); //parameterized constructor`
		Also add relevant destructor.
		In `main()` function, create three Student objects '`ram`', '`rahim`' and '`joseph`'. Observe behaviour of constructor & destructor.

2	*	In the `Vector` class, add following constructors: `Vector();` //default constructor. So no elements in Vector `Vector(int countOfElements);` //Parameterized constructor Also add relevant destructor to release (delete) the dynamically allocated memory. Demonstrate usage of these three special methods.
3	*	In the `Matrix` class, add following constructors: `Matrix ();` //default constructor. So no elements in Matrix `Matrix (int countOfRows, int countOfColumns);` //Parameterized constructor Also add relevant destructor to release (delete) the dynamically allocated memory. Demonstrate usage of these three special methods.
4	*	In the `String` class, add following constructors: `String();` //default constructor. So, null string `String(char* s);` //Parameterized constructor Also add relevant destructor to release (delete) the dynamically allocated memory. Demonstrate usage of these three special methods.
5		Add default constructor & parameterized constructor to the `Complex` class.
6		Add default constructor & parameterized constructor to the `Point` class.
7	*	Add default constructor & parameterized constructor to the Date class.
8	*	Add copy constructor to `Student` class. Demonstrate usage of the same. Observe program behaviour.
9	*	Add copy constructor to `Vector` class. Demonstrate usage of the same. Observe program behaviour.
10	*	Add copy constructor to `String` class. Demonstrate usage of the same. Observe program behaviour.
11		Add copy constructor to `Matrix` class. Demonstrate usage of the same. Observe program behaviour.