The objective of this lab is to:

1. Understand and practice operators overloading as non-member functions.
2. Practice good coding conventions e.g commenting, meaningful variable and functions names, properly indented and modular code.

Instructions!

1. This is a **graded** lab, you are strictly **NOT** allowed to discuss your solutions with your fellow colleagues, even not allowed asking how is he/she is doing, it may result in negative marking. You can **ONLY** discuss with your TAs or with me.
3. Strictly follow good coding conventions (commenting, meaningful variable and functions names, properly indented and modular code.
4. Save your work frequently. Make a habit of pressing **CTRL+S** after every line of code you write.

Task 01: [15 Marks]

Implement class PhoneCall having data members callerNo, duration in minutes and rate charged per minute.

```
class PhoneCall
{
private:
	string callerNo;   // Phone number.
	int duration;      // Duration of the call in minutes.
	float rateCharged; // rate charged per minute.
public:

	// functions ...
};
```

Implement getter/setter functions and appropriate constructors. Also,

- Overload extraction operator " << " to output a PhoneCall in bellow format:

Caller No: 03001231231 Duration: 3 minutes Rate Charged: 1.45 / minutes

- Overload insertion operator ">>" to take input with appropriate messages.
- overload the == operator to compare two PhoneCalls. Consider one phone call equals to another if both calls are placed to the same number.
- Overload > operator to compare two PhoneCalls, consider one phone call is greater to another if one call's duration is greater than other call's duration.
- Overload + operator to add duration of two calls and return int.
- Overload conversion operator int that should convert a PhoneCall to int by returning duration of the call.
- Overload conversion operator double that should convert a PhoneCall to double by returning rateCharged of the call.

Write main function to create objects of Phone Call and test all functions.

Task 02: [15 Marks]

Create class ScoreKeeper that tracks the scores a student in a course. Include fields for the name of the course, an integer that holds the number of scores (i.e. 2 Quiz + 1 Assignment means 3 scores) and an integer pointer that points to a list of the student's scores on tests and assignments in a class.

```
class ScoreKeeper
{
private:
	string course;		// name of the course
	int scoresCount;	// an integer that holds the number of scores.
	Int* scores;		// pointer that points to a list of the student's scores
public:

	// functions ...
};
```

- Add a constructor that accepts the course name and number of scored items and then prompts the user for the individual scores. Each score must be a value from 0 to 100; if the score is too high or too low, re-prompt the user for a valid score.
- Overload an insertion operator that displays data such as, first display course name and no of scores, in second line show all scores separated by space.
- overload an = operator that assigns one ScoreKeeper to another.
- Overload subscript operator [ ] to access list of scores as an array.

Write a *main( )* function to exhibit each of the above functionalities.