North South University
CSE-225L(Data Structures & Algorithm)
Fall - 2018
Lab-01 (Course Policies; Objects & Classes in C++)

Course Details:	**Pre-requisites**:
Course: CSE225L Section 5 & 6 **Class Time-slot:** **Section 5**: MW (1120 AM : 1250 AM, LIB605) **Section 6**: MW (0240 PM : 0410 PM, LIB607) **Facebook Groups:** **Section 5:** NSU-CSE-225L.5(Fall-2018) **Section 6:** NSU-CSE-225L.6(Fall-2018)	CSE-115, CSE-215 **Instructor:** Md. Nazmul Hossain **Email** : nazmul.hossain@northsouth.edu **Phone:** 01731492836 **Room No**. : LIB600 (C4) **Office Hours:** **Sunday & Tuesday: (0100PM-0410PM)** **Monday & Wednesday: (0100PM-0230PM)** **Thursday: (0100PM-0230PM)**

Class and Course Policy:

1) **DO NOT EVER BE LATE IN THE CLASS.**
2) Join your respective section's Facebook group (names given above). All important class related updates/ notices will be posted there.
3) Starting from the third lab class and onwards, there will be **graded practice session** in each class which will be the basis for determining your final **Class Performance** score. Also, starting from first lab class, every single class will carry attendance mark.
4) All of you **MUST** have your own official **"northsouth.edu"** email account which you will need to access the shared Google sheet (link will be posted on the Facebook group) where your practice session and other marks will be continuously updated. **No "northsouth.edu" account, no access to your marks.**
5) **No make-up practice session/ midterm exam/ final exam** will be allowed unless it's a medical/ family emergency.
    - In both the cases, you'll need explicit written permission from the faculty of the associated theory class to sit for any kind of make-up exams.
    - Also, in case of make-up exams due to medical reasons - you'll need to provide valid medical documents (prescriptions and/ or test reports, etc.) along with your application.

'Academic Honesty' policies:

- Honest academic behavior will be of utmost importance.
- Any form of **dishonest academic behaviour** (copying of source codes, cheating during exams/ lab-evaluations) **will be very harshly dealt**.
- In both the cases of lab evaluations and lab exams, **the person copying** and **the person letting copy his/ her code**, will be **awarded zero as their lab evaluation/ exam score** during that class/ exam. Suspiciously similar code structure/ variable names/ solving techniques will be considered 'copy' works.

Percentage Breakdown of Grade Calculation:

Topic	Allotted Percentage
Attendance	5%
Class Performance (Average of All Class Performances)	10%
Assignment (Average of All Assignments)	10%
Midterm	15%
Presentation	60%

How to write a simple class in C++:

In C++, the following is the general format for a class declaration and definition:

class class-name{

access-specifier:
respective member variables and functions

another access-specifier:
respective member variables and functions

};

Here, access-modifiers can be: public/ private/ protected (just like in JAVA). **By default**, functions and data declared within a C++ class are private.

Suppose, **in JAVA**, you have written the following class named **DynamicArray-**

```
public class DynamicArray{

private int[] data;

public DynamicArray(int size)
{
      data = new int[size];
}

public void insertItem(int index, int item)
{
      data[index] = item;
}

public int getItem(int index)
{
      return data[index];
}

}
```

Now, in the main method, you create an object of that above class like this:

```
public static void main(String[] args)
{
  DynamicArray d = new DynamicArray(10);
}
```

Now, if you convert the above JAVA class into a C++ class, it'll consist of the following different parts:

- The first part is the 'header' file **(**with the file extension **.h)** which will contain only the declarations of all the class variables and class functions, no implementation here.

dynamicarray.h

```
#ifndef DYNAMICARRAY_H_INCLUDED
#define DYNAMICARRAY_H_INCLUDED

class DynamicArray{

private:
      int* data;
public:
      DynamicArray(int);
      ~DynamicArray();
      void insertItem(int, int);
      int getItem(int);
};

#endif
```

- The second part is the cpp file **(**with the file extension **.cpp)** which will contain only the **definitions** of all the class variables and class functions **'declared'** in the previous class header file. You **MUST** have to **include** the header file inside this cpp file.

dynamicarray.cpp

```
#include "dynamicarray.h"
DynamicArray::DynamicArray(int size)
{
	data = new int[size];
}
void DynamicArray::insertItem(int index,
	int item)
{
	data[index] = item;
}
```

```
int DynamicArray::getItem(int index)
{
	return data[index];
}

DynamicArray::~DynamicArray()
{
	delete[] data;
}
```

Now, in the main c++ file (also sometimes called the **driver file**) named **main.cpp**, you create and manipulate a DynamicArray class object as described below:

main.cpp

```
#include "dynamicarray.h"
#include <iostream>
using namespace std;

int main()
{
	// Prompting the user to enter the size of the array
	cout<<"Enter the size of the array: "<<endl;
	int size;
	// Taking the input from the user and assigning that value to the int variable
	// named size
	cin>>size;

	// Creating the DynamicArray class object with the specified size
	DynamicArray d(size);

// Taking inputs from the user and inserting them inside the DynamicArray object
// created above
	int temp;

	for(int i=0;i<size;i++)
	{
		cout<< "Enter value to be inserted at index = "<<i<<endl;
		cin>>temp;
		d.insertItem(i, temp);
	}

	// Printing all the integer values saved in the DynamicArray class object
	cout<< "The values stored are: ";

	int temp2;

	for(int i=0;i<size;i++)
	{
		temp2 = d.getItem(i);
		cout<< "Index = "<<i; cout<< ", Value = "<<temp2<<endl;

	}

	return 0;
}
```