The objective of this lab is to:

Practice string functions, structures, nested structures, pointer to objects and array of objects.

Instructions!

1. Please follow the dress code before coming to the lab. Keep your student identity cards with you.
2. This is an individual 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.
1. Strictly follow good coding conventions (commenting, meaningful variable and functions names, properly indented and modular code.
2. Save your work frequently. Make a habit of pressing **CTRL+S** after every line of code you write.
3. Beware of **memory leaks** and **dangling pointers**.

Task 01: [15 Marks]

1. Create a structure *InventoryItem* that contains four data members: an item code (*int*), item name (*string*), cost (*double*), selling price (*double*).
   Your program should have an array of at least 10 Inventory items. Take the name and cost of each inventory item from the user in a function *insertItem*. Item code should be an auto-generated integer (you may use rand( ) function starting from anywhere, for example, 3123 or generate random numbers between a range), and selling price should be computed on the basis of 40% profit on cost. Create a function *displayStock* that should display all the items on the console in a clear readable format (better be in tabular format). You should implement the following two functions for this task:

```
a) InventoryItem insertItem();
b) void displayStock( const InventoryItem* );
```

2. Modify the above task such that the item code is generated as a combination of characters and numbers. Change the data type of item code to *string*. To generate an item code, you should combine the first three characters of the item name (in upper case) with a running integer, both separated with an underscore. For example, if the first item to be stocked in, is a computer then its code should be COM_31203, and the second item is a hard disk then its code should be HAR_31204, and so on.
   Hint: Use appropriate string functions to get substring, to convert string in upper case and string concatenations.

Task 02: [10 Marks]

Suppose you run a carpet company and you want to write an application that can estimate the cost of carpeting in a house on per-room bases by calculating the price of carpeting for each rectangular room in the house. To calculate the price, you multiply the area of the floor (width times length) by the price per square foot of carpet. For example, the area of the floor that is 12 feet long and 10 feet wide is 120 square feet. To cover that floor with carpet costs Rs. 18 per square foot would cost Rs. 2,160. (12 × 10 × 18 = 2,160.)

First, you should create a struct named *RoomDimension* that has two *FeetInches* (both the structs are given below) objects as data members: one for the length of the room and one for the width.

Write code in main( ) that asks the user to enter the number of rooms in your house. Create a dynamic array of the size equal to the number of rooms in the house to store the RoomDimension of each room. Then, ask the dimensions of each room in the house from the user and populate the array with user-given data.

Your program should display the total cost of the carpeting in the house. The user can also see the cost of carpeting for each room separately. Create a menu based program to offer both services to the user. Create a separate function to calculate the cost of carpet for each room. You may use the following function prototype and struct definitions.

```
int computeTheCarpetingCost(const int& RoomDimension);
```

```
struct FeetInches
{
       int feet;
       int inches;
};
```

```
struct RoomDimension
{
       FeetInches length;
       FeetInches Width;
};
```