East West University
Department of Computer Science and Engineering

CSE207 – Data Structures: LAB 01
Course Instructor: Ms. Tanni Mittra

Dynamic Memory Allocation and Class

A. Class and DMA

1. A grocery shop owner wants to store the information about the products that he has in the stock. A product has its unique id, name, brand name, type (for example food, cosmetic, electronic etc.), quantity and price of each unit. **First** you have to design a **class** with appropriate entry according to the problem specification. You should have come up with something like this:

```
class product
      {
            Private:
                  int id;
                  string name;
            ........
      };
```

   a. Now create an array of the objects and let the user to decide how many products info he/she wants to store. Then store the information using the created array of objects. After that display the name of the products and their prices whose prices are greater than 40.
   b. Repeat the process of (a) using dynamic memory allocation.

2. Consider the problem of the previous question and solve the following using dynamic memory allocation:
   a. User will input the brand name and you have to display every product info of that particular brand.
   b. User will input the type of product and you have to calculate the total asset of that particular type. `(qty*price)`
   c. Calculate the total asset of the grocery shop.

3. Consider a class having two numbers `range1` and `range2`. `range1` must be smaller than `range2`. The structure also has a `counter` variable and an integer type array `num`. You have to design a program which will generate all the prime numbers in the range of `range1` to `range2` and store them into the array of this structure. You also have to calculate the number of prime numbers in the given range and store that into the `counter` variable of the structure.
   After creating and preparing the structure according to the above-mentioned criterion, you have to print that structure with appropriate messages. You have to print all the prime numbers in that range using pointer (direct array print is not allowed). Use dynamic memory allocation.

B. **Class**

- A class is a blueprint or abstraction of same type of objects which possess both data and methods/functions. This term is known as encapsulation.
- Here is an example of a C++ class:

```
class Rectangle
{
private:
    double height;
    double width;
public:
    Rectangle() {};
    Rectangle(double height,double width)
    {
        this->height=height;
        this->width=width;
    }
    void setheight(double h)
    {
        height=h;
    }
    void setwidth(double w)
    {
        width=w;
    }
    double getheight()
    {
        return height;
    }
    double getwidth()
    {
        return width;
    }
    void displayArea()
    {
        double area=height*width;
        cout <<" Area: " << area;
    }

};
```

- **Constructor** has same name as the class itself Constructors don't have return type A constructor is automatically called when an object is created. `Rectangle()` and `Rectangle(double height,double width)` are the constructors. Recall the concept of method/function overloading in Object Oriented Programming.

- Getters and Setters allow you to effectively protect your data. `setheight(double h)` is a setter and `getwidth()` is a getter. This is a technique used greatly when creating classes. For each variable, a get method will return its value and a set method will set the value.

- Write the main function after writing the given class. In main do the following:

i. Create normal object of `Rectangle` class, initialize after taking user inputs and display the area.
ii. Create another object so that you can allocate memory for that object dynamically and demonstrate use of that object.
iii. Create an array of objects of `Rectangle` class, initialize the whole array taking user inputs. Then display area of each rectangle in a separate loop.
iv. Create a pointer object of `Rectangle` class. With help of that single pointer object repeat the tasks of (iii). Do not use array.