National University of Computer & Emerging Sciences, Karachi
Computer Science Department
Spring 2021, Lab Manual - 03

Course Code: CL-217	Course : Object Oriented Programming Lab
Instructor(s) :	Nida Munawar, Abeer Gouhar, Romasha Khurshid, M. Fahim, Sohail Afzal, Qaiser Abbas, Ali Fatmi

Contents:

Introduction to Classes & Objects

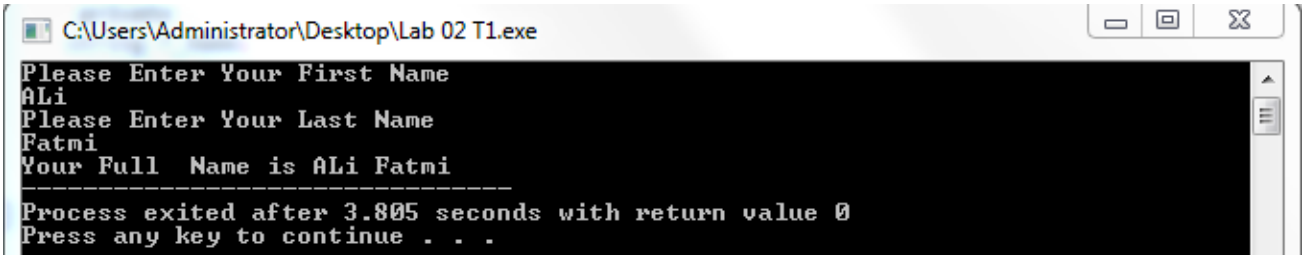
- A ***class*** *is a definition of objects of the same kind*. In other words, a *class* is a blueprint, template, or prototype that defines and describes the *static attributes* and *dynamic behaviors* common to all objects of the same kind.
- A **class** can be visualized as a three-compartment box, as illustrated:
  - ***Classname*** (or identifier): identifies the class.
  - ***Data Members*** or ***Variables*** (or *attributes*, *states*, *fields*): contains the *static attributes* of the class.
  - ***Member Functions*** (or *methods*, *behaviors*, *operations*): contains the *dynamic operations* of the class.
- An **Object** is an instance of a Class. When a class is defined, no memory is allocated but when it is instantiated (i.e. an object is created) memory is allocated. When a class is defined, only the specification for the object is defined; no memory or storage is allocated. To use the data and access functions defined in the class, you need to create objects.

Sample C++ Code:

Code#1

```
#include<iostream>
using namespace std;

class Studnet
{
   private :
  string F_Name;
   string L_Name;

  public:
```

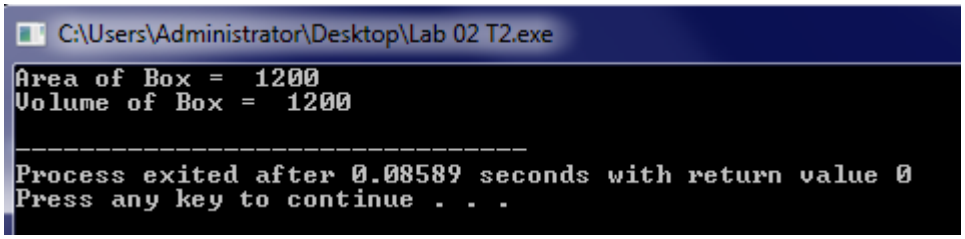
```
    void input_value()
    {
      cout << "Please Enter Your First Name\n";
      cin >> F_Name;
      cout << "Please Enter Your Last Name \n";
      cin >> L_Name;
    }

    void output_value()
    {
      cout << "Your Full  Name is "<<F_Name<<" " <<L_Name;

    }
};

main()
{
  Studnet object;

  object.input_value();
  object.output_value();

  //object.variable;  Will produce an error because variable is private

  return 0;
}
```



Code#2

```
#include <iostream>
using namespace std;

class Box {

  public:
   double Lenght;
   double Breadth;
```

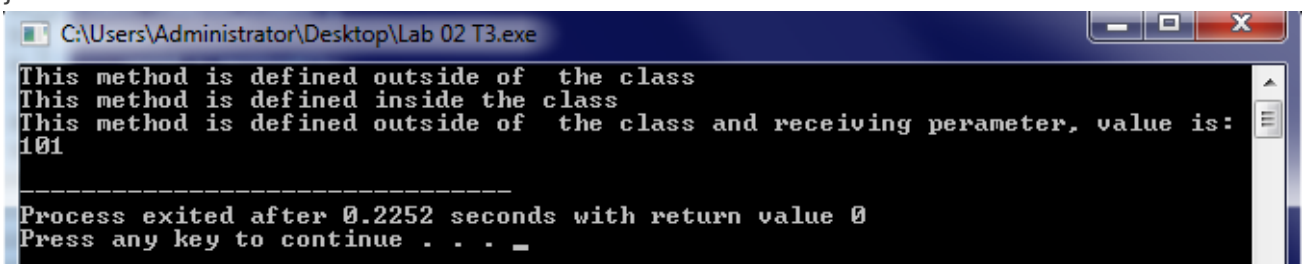
```
    double Height;

    double Area() {
      return Lenght * Breadth;
    }

    double Volume() {
      return Lenght * Breadth * Height;
    }
};

int main() {


    Box obj;

    obj.Lenght = 30;
    obj.Breadth = 40;
    obj.Height = 60;


    cout << "Area of Box =  " << obj.Area() << endl;
    cout << "Volume of Box =  " << obj.Area() << endl;

    return 0;
}
```



Code#3

```
#include <iostream>
using namespace std;

class Sample_Class {
public:
   void o_method();
   void o_method2(int value);

   void i_method()
```

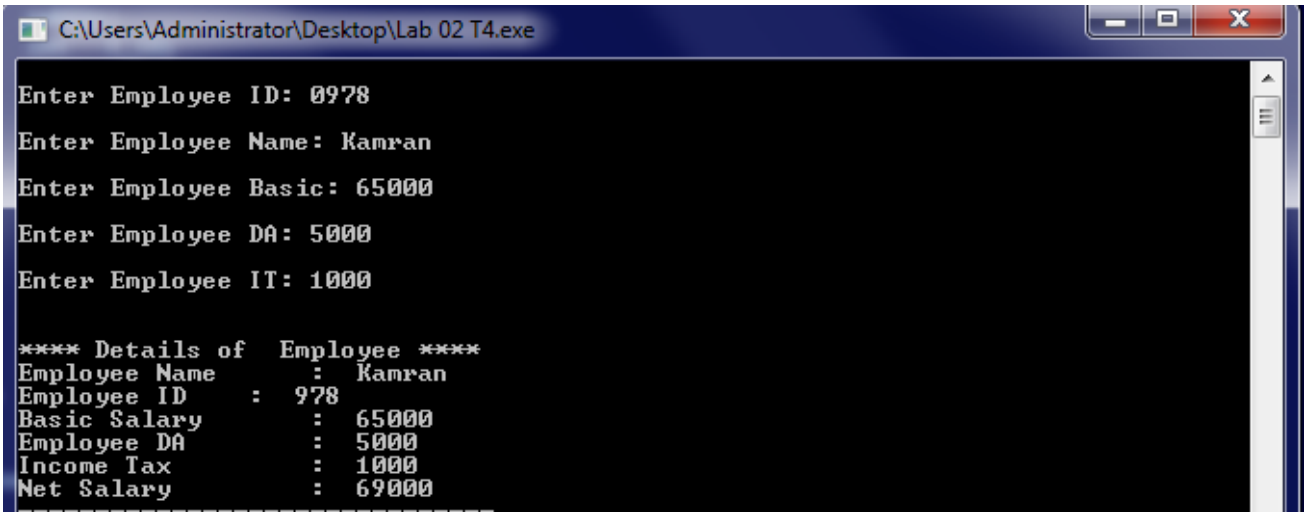
```
{
   cout << "This method is defined inside the class\n";
}

};


void Sample_Class::o_method()
{
   cout << "This method is defined outside of  the class\n";
}




void Sample_Class::o_method2(int number)
{
   cout << "This method is defined outside of  the class and receiving perameter, value is: " << number << "\n";
}

int main()
{

   Sample_Class obj;
   obj.o_method();
   obj.i_method();
   obj.o_method2(101);

   return 0;
}
```



Code#4

```
#include <iostream>
using namespace std;

class employee
{
        public:
        int   E_id;
        string  E_name;
        float E_basic;
        float E_da;
        float E_it;
        float E_net_sel;

        public:


                float find_net_salary(float basic, float da, float it);
                void show_emp_details();
};


float employee :: find_net_salary(float basic, float da, float it)
{
   return (basic+da)-it;
}

void employee :: show_emp_details()
{
        cout<<"\n\n**** Details of  Employee ****";
        cout<<"\nEmployee Name      :  "<<E_name;
        cout<<"\nEmployee ID    :  "<<E_id;
        cout<<"\nBasic Salary       :  "<<E_basic;
        cout<<"\nEmployee DA        :  "<<E_da;
        cout<<"\nIncome Tax         :  "<<E_it;
        float net_salary=find_net_salary(E_basic, E_da, E_it);
        cout<<"\nNet Salary         :  "<<net_salary;
        cout<<"\n-----------------------------\n\n";
}


int main()
```

```
{
   employee emp;
   cout<<"\nEnter Employee ID: ";
        cin>>emp.E_id;
        cout<<"\nEnter Employee Name: ";
        cin>>emp.E_name;
        cout<<"\nEnter Employee Basic: ";
        cin>>emp.E_basic;
        cout<<"\nEnter Employee DA: ";
        cin>>emp.E_da;
        cout<<"\nEnter Employee IT: ";
        cin>>emp.E_it;

   emp.show_emp_details();


   return 0;
}
```

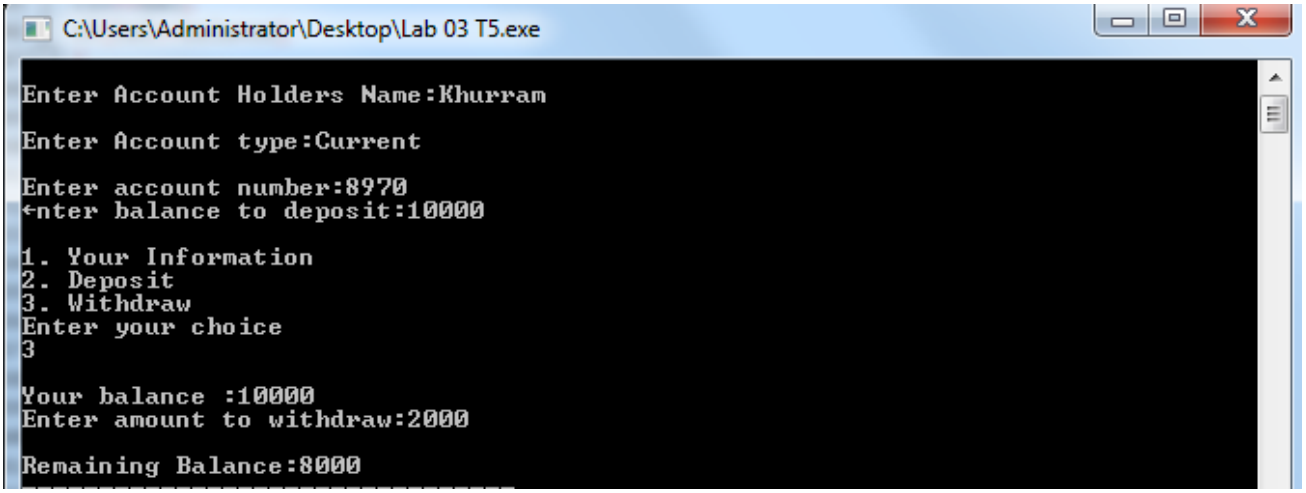


Code#5

```
#include<iostream>
using namespace std;

class Bank
{


public:
string name;
string account_type;
int account_number;
int balance;

//member functions of the class Bank
// initialize function to initialize data members

void initialize()
{
cout<<"\nEnter Account Holders Name:";
cin>>name;
cout<<"\nEnter Account type:";
cin>>account_type;
cout<<"\nEnter account number:";
cin>>account_number;
cout<<"\Enter balance to deposit:";
cin>>balance;
}


void deposit()
{
int bal;
cout<<"\nEnter the amout to deposit:";
cin>>bal;
balance+=bal;
cout<<"\nAmount deposited successfuly\nYour New Balance:"<<balance;
}

//check() function to withdraw amount and check remaining balance

void check()
```

```
{
int bal;
cout<<"\nYour balance :"<<balance<<"\nEnter amount to withdraw:";
cin>>bal;
if(bal<=balance)
{
balance-=bal;
cout<<"\nRemaining Balance:"<<balance;
}
else
{
exit(0);
}
}

//display function to display user information
void display()
{
cout<<"\nName :";
cin>>name;
cout<<"\nBalance :"<<balance;
}
};

int  main()
{
int i;


Bank bk;
bk.initialize();
cout<<"\n1. Your Information\n2. Deposit\n3. Withdraw\nEnter your choice\n";
cin>>i;
if(i==1)
{
bk.display();
}
else if(i==2)
{
bk.deposit();
}
else if(i==3)
```

```
{
bk.check();
}
return 0;
}
```



LAB TASKS:

Task - 01:

Create a class User with two public fields: int Age and string Name. In the Main method, create an object of class User and set Name to "Teo" and Age to 24. Then, output to the screen: "My name is {Name} and I'm {Age} years old." using object fields for Name and Age.

Task - 02:

Create a class called Date that includes three pieces of information as instance variables—a month (typeint), a day (typeint) and a year (typeint). Provide a method displayDate that displays the month, day and year separated by forward slashes(/). Write a test application named DateTest that demonstrates classDate's capabilities.

Task - 03:

We are prototyping a robot that refills glasses during dinner. Every glass holds 200 milliliters. During dinner, people either drink water or juice, and as soon as there is less than 100 ml left in the glass, the robot refills it back to 200 ml. Create a class Glass with one public int field LiquidLevel and methods public Drink(int milliliters) that takes the amount of liquid that a person drank and public Refill() that refills the glass to be 200 ml full. Both methods should not return any value. Initially set LiquidLevel to 200. In the Main method create an object of class Glass and read

commands from the screen until the user terminates the program (see next). Don't forget to refill the glass when needed!

Task - 04:

Create a class called Employee that includes three pieces of information as instance variables—a first name (type String), a last name (type String) and a monthly salary (double). If the monthly salary is not positive, set it to 0.0. Write a test application named EmployeeTest that demonstrates class Employee's capabilities. Create two Employee objects and display each object's yearly salary. Then give each Employee a 10% raise and display each Employee's yearly salary again.

Task - 05:

. Create a class called Book to represent a book. A Book should include four pieces of information as instance variables-a book name, an ISBN number, an author name and a publisher. Provide methods (query method) for each instance variable. Inaddition, provide a method named getBookInfo that returns the description of the book as a String (the description should include all the information about the book). You should use this keyword in member methods and constructor. Write a test application named BookTest to create an array of object for 5 elements for class Book to demonstrate the class Book's capabilities.