Lab Sheet 3
Concept of Objects and Classes

Objects

Object consists of the data and the functions that operate on the data. Object data is usually private: no functions other than those within the same object can access it. On the other hand, the functions within the object are usually public. They can be accessed by the other functions in the program including the functions in the other objects. So the function in one object can access the data in the other object through the function of that object.

Classes

In oops. Objects are variables of classes. It is a specification for any number of objects based on that class. Class itself is a template (data type) for objects. When we actually define the object, of a class we are requesting to allocate memory for that object to hold its data.

Comments in C++

In C++, a new symbol for comment '//' (double slash) is introduced and is called one-line comment.

```
// this is an example of comment illustration

stdno=48;    //no of student in 059/bct batch
```

C comment symbols '/*' and '*/' are also valid and can be used for multi-line comments.

```
/*this is an example

    illustrating the multi-

    line comment in C++ */
```

Output Statement

The statement

```
cout << "This is the first lab in C++";
```

cause the string "This is the first lab in C++" to be displayed on the screen. Here cout is the predefined object that represents the standard output stream in C++.

The operator '<<', insertion operator, has a simple interface, i.e it is not necessary to specify the data type of the variable on its right. The insertion operator automatically works with any type of the variable. Another advantage of this is that user-defined data types can also be used with this operator, which is not possible with printf( ) function.

Input Statement

The following statement reads the value from the keyboard and places it in a variable size.

```
cin >> size;
```

cin identifier represents the standard input device. The >> operator (extraction operator) takes the input from the device. The function is smart enough to interpret the input according to the data type of the variable that holds the value. If size is an integer and the user types "25", the integer value 25 will be placed in size. If the size is a string, the string "25" will be placed in it.

Data Member

The data items within a class are called data members. There can be any number of data members in a class. Normally, data members follow the keyword private, so they can be accessed from within the class but not from outside. That is why we say that oops have feature of data hiding. So, it is safe from accidental alteration.

```
class demo

{

  private:

     int rollno;           //member data

     float score;         //member data

  public:

     setdata( int     //member function
     rl, float sc)

     {rollno=rl; score=sc;}

     setdata( )
```

```
{
  cout<<"Enter the roll no: \n";
  cin>>rollno;
  cout<<"Score: \n";
  cin>>score;
}
showdata( )     //member function
{
  cout<<"\Roll No:    " <<rollno<<"has scored
  "<<score<<endl;
}
```

```
};
```

above class contains two data items: rollno and score. Here rollno is of type int and score is of type float.

Member Function

Member functions are functions that are included within a class. In above example there are two member functions in class demo: setdata( ) and showdata( ). Each function can have one or more statements. The functions setdata( ) and showdata( ) follow the keyword public which means that they can be accessed from outside the class. It is also possible to declare a function within a class and define it elsewhere.

The class demo can be used as follows

```
int main()

{

  demo d1,d3;

  d1.setdata(12,10.4);

  d3.setdata( );

  d1.showdata( );

  d3.showdata( );

  return 0;

}
```

Defining the Objects

Here the d1 and d3 are defined as objects of class demo. Defining the object is similar to defining a variable of any data type. Space is set-aside for it. Defining the object is creating them i.e instance of the class is created. In general objects in the

programs represent physical objects: things that can be felt or seen, for example circle, person, car etc

Calling Member Functions

Member functions are called as follows

```
d1.setdata(12,10.4);

d3.setdata( );

d1.showdata( );

d3.showdata( );
```

This syntax is used to call a member function that is associated with a specific object. Because of the demo class, it must always be called in connection with an object of this class. Here the first member function is called by passing the values while the second is called without passing any value.

Exercises

1. **Write a simple program that convert the temperature in degree Celsius to degree Fahrenheit and vice versa using the basic concept of class and object. Make separate class for Centigrade and Fahrenheit which will have the private member to hold the temperature value and make conversion functions in each class for conversion from one to other. For example you will have function toFahrenheit() in class Celsius that converts to Fahrenheit scale and returns the value.**

```
#include <iostream>
using namespace std;
class fahrenheit{
  float fah;
  public:
  float getfahrenheit()
  {
  cout<<"Enter 1st tem. in F.\t"; cin>>fah;
  return fah;
  }
  float calculatec()
 {
  return ((fah-32)*100)/180;
 }}fahr;
 class celcius{
 float cel;
 public:
 float getcelcius()
 {
  cout<<"Enter 2nd tem. in C. \t"; cin>>cel; return
cel;
 }
 float calculatef()
 {
  return (cel*180)/100+32;
 }}celc;
int main()
{
 float fah=fahr.getfahrenheit(); float
cel=celc.getcelcius();
```

```
 cout<<fah<<" F to Celcius is "<<fahr.calculatec()<<"
C."<<endl;
 cout<<cel<<" C to Fahrenheit is
"<<celc.calculatef()<<" F."<<endl;
 return 0;
}
```

2. Assume that you want to check whether the number is prime or not. Write a program that asks for a number repeatedly. When it finishes the calculation the program asks if the user wants to do another calculation. The response can be 'y' or 'n'. Don't forget to use the object class concept.

```
#include <iostream>
using namespace std;
class prime{
  int n;
  public:
  void get_input()
  {
   cout<<"Enter test number.\t";
   cin>>n;
  }
  void check_prime()
  {
   int i,c=0;
   for(i=2;i<n;i++)
    {
     if (n%i==0) c++;
    }
   if(c==0) cout<<"PRIME!!!"<<endl; else cout<<"NOT
PRIME!!!"<<endl;
  }
  }p;
int main()
  {
   char a='y';
   do{
   p.get_input();
   p.check_prime();
   cout<<"Do you want to enter another
number(y/n)?"<<endl;
```

```
   cin>>a;
   }while (a=='y');
  return 0;
}
```

3. Create a class called carpark that has int data member for car id, int data member for charge/hour and float data member for time. Set the data and show the charges and parked hours of corresponding car id. Make two member functions for setting and showing the data. Member function should be called from other functions.

```
#include <iostream>
using namespace std;
class carpark{
  int carid;
  int cost1,cost;
  float time;
  public:
  void input()
  {
    cout<<"Enter car id."<<endl;
    cin>>carid;
    cout<<"Enter cost per hour."<<endl; cin>>cost1;
    cout<<"Enter parking time in hours."<<endl;
    cin>>time;
  }
 void calculation()
 {
 cost=cost1*time;
 }
 void display()
 {
  cout<<"Parking cost of car "<<carid<<" for parking
"<<time<<" hours as per cost "<<cost1<<" per hour is
Rs "<<cost<<"."<<endl;
 }
 }park;
int main()
 {
  park.input();
  park.calculation();
  park.display();
  return 0;
 }
```

4. Write a program with classes to represent circle, rectangle and triangle. Each classes should have data members to represent the actual objects and member functions to read and display objects, find perimeter and area of the objects and other useful functions. Use the classes to create objects in your program.

```
#define pi 3.1415
#include <iostream>
#include <cmath>
using namespace std;
class geo{
float r,l,b,l1,l2,l3,ac,pc,ar,pr,atr,pt;
public:
 void read_geometry()
 {
  cout<<"Enter radius of circle."<<endl;
  cin>>r;
  cout<<"Enter length of rectangle."<<endl;
   cin>>l;
  cout<<"Enter breadth of rectangle."<<endl;
  cin>>b;
  cout<<"Enter lenth of triangle 1st side."<<endl;
  cin>>l1;
  cout<<"Enter lenth of triangle 2nd side."<<endl;
  cin>>l2;
  cout<<"Enter lenth of triangle 3rd side."<<endl;
  cin>>l3;
 }
 void calculation()
 {
  ac=pi*r*r;
  pc=2*pi*r;
   float s=(l1+l2+l3)/2; atr=sqrt(s*(s-l1)*(s-l2)*(s-
l3));
  pt=2*s;
  ar=2*(l+b);
 }
 void display_geometry()
 {
  cout<<"Area of circle= "<<ac<<endl;
  cout<<"Perimeter of circle= "<<pc<<endl;
  cout<<"Area of square= "<<l*b<<endl;
  cout<<"Perimeter of square= "<<ar<<endl;
```

```
  cout<<"Area of traingle= "<<atr<<endl;
  cout<<"Perimeter of traingle= "<<pt<<endl;
 }
}read;
int main()
{
 read.read_geometry();
 read.calculation();
 read.display_geometry();
 return 0;
}

#include<iostream> //OR
#include<math.h>
#define PI 3.14156
using namespace std;
class circle
{
private:
    float radius;
public:
    void data (float x)
    {
        radius=x;
    }
    void calculations();
};
class rectangle
{
private:
    float length,breadth;
public:
    void data (float x,float y)
    {
        length=x;
        breadth=y;
    }
    void calculations();
};
class triangle
{
private:
    float l1,l2,l3;
public:
    void data (float x,float y,float z)
```

```
    {

        l1=x;
        l2=y;
        l3=z;
     }
    void calculations();
};
int main()
{
    class circle c;
    class rectangle rec;
    class triangle t;
    float r,l,b,l1,l2,l3;
    cout<<"For circle:"<<endl<<"Enter radius:\t";
    cin>>r;
            cout<<"\n\nFor   rectangle:"<<endl<<"Enter
length:\t";
    cin>>l;
    cout<<"\nEnter breadth:\t";
    cin>>b;
     cout<<"\n\nFor triangle:"<<endl<<"Enter length of
first side:\t";
    cin>>l1;
    cout<<"\nEnter length of second side:\t";
    cin>>l2;
   cout<<"\nEnter length of third side:\t";
    cin>>l3;
    c.data(r);
    c.calculations();
    rec.data(l,b);
    rec.calculations();
    t.data(l1,l2,l3);
    t.calculations();
    return 0;
}
void circle::calculations()
{
                                                cout<<"\n\nFor
circle:"<<endl<<"Radius:\t"<<radius<<endl;
    cout<<"Perimeter:\t"<<(2*PI*radius)<<endl;
    cout<<"Area:\t"<<(PI*radius*radius)<<endl;
}
void rectangle::calculations()
{
```

```
                                                cout<<"\n\nFor
rectangle:"<<endl<<"Length=:\t"<<length<<endl;
    cout<<"Breadth=:\t"<<breadth<<endl;
    cout<<"Perimeter=:\t"<<2*(length+breadth)<<endl;
    cout<<"Area=:\t"<<(length*breadth)<<endl;
}
void triangle::calculations()
{
    float area,p;
    p=(l1+l2+l3)/2;
    area=sqrt(p*(p-l1)*(p-l2)*(p-l3));
     cout<<"\n\nFor triangle:"<<endl<<"Length of first
side:\t"<<l1<<endl;
    cout<<"Length of second side:\t"<<l2<<endl;
    cout<<"Length of third side:\t"<<l3<<endl;
    cout<<"Perimeter:\t"<<2*p<<endl;
    cout<<"Area=:\t"<<area<<endl;
}
```

5. Assume that an object represents an employee report that contains the information like employee id, total bonus, total overtime in a particular year. Use array of objects to represent *n*employees' reports. Write a program that displays report. Use setpara() member function to set report attributes by passing the arguments and member function displayreport() to show the reports according to parameter passed. Display the report in following format.

```
Employee with ... ... ... has
received Rs ... ... ...as
bonus and
had worked ... ... ... hours as a over time in
year ... ... ...
```

```
#include <iostream>
using namespace std;
class employee{
int id,year;
float bonus,overtime;
public:
  void setpara()
 {
  cout<<"Enter employee id.\t";
```

```
  cin>>id;
  cout<<"Enter employee bonus.\t"; cin>>bonus;
  cout<<"Enter employee overtime in hours.\t";
cin>>overtime;
  cout<<"Enter year.\t"; cin>>year;
 }
 void display_report()
 {
  cout<<endl<<"Employee with "<<id<<" has received Rs
"<<bonus<<" as "<<endl;
  cout<<"bonus and"<<endl;
  cout<<"had worked "<<overtime<<" hours as overtime
in "<<year<<" year."<<endl;
 }
 };
int main()
 {
 int n,i;
 cout<<"Enter employee number"<<endl; cin>>n;
employee e[n];
 for(i=0;i<n;i++)
 {
  cout<<endl<<"For "<<i+1<<" employee."<<endl;
e[i].setpara();
 }
 for(i=0;i<n;i++)
 {
  e[i].display_report();
 }
  return 0;
}

#include<iostream> //OR
using namespace std;
class emp_report
{
    int emp_id,year;
    float total_bonus,total_overtime;
public:
    void setpara();
    void displayreport(emp_report emp_data[],int
no_of_emp);
};
int main()
```

```
{
    int no_of_emp;
    cout<<"Enter total number of employees:\t";
    cin>>no_of_emp;
    emp_report emp[no_of_emp],disp_report;
    for(int i=0;i<no_of_emp;i++)
    {
        cout<<"\nfor emp:"<<i+1;
        emp[i].setpara();
    }
    disp_report.displayreport(emp,no_of_emp);
    return 0;
}
void emp_report::setpara ()
{
    cout<<"\nEnter your employee id:\t";
    cin>>emp_id;
    cout<<"\nEnter total amount of bonus
received:\t";
    cin>>total_bonus;
    cout<<"\nEnter total amount of overtime
worked:\t";
    cin>>total_overtime;
    cout<<"\nEnter the working year:\t";
    cin>>year;
}
void emp_report::displayreport(emp_report
emp_data[],int no_of_emp)
{
    for(int i=0;i<no_of_emp;i++)
    {
        cout<<"\nEmployee with
"<<emp_data[i].emp_id<<" has received Rs
"<<emp_data[i].total_bonus<<" and had worked
"<<emp_data[i].total_overtime<<" hours as a overtime
in year "<<emp_data[i].year;
    }
}
```