FAST National University of Computer and Emerging Sciences Peshawar

OOP (Lab-10)

C++ (Classes and Objects)

Computer Instructor: Engr. Khuram Shahzad

DEPARTMENT OF COMPUTER SCIENCE

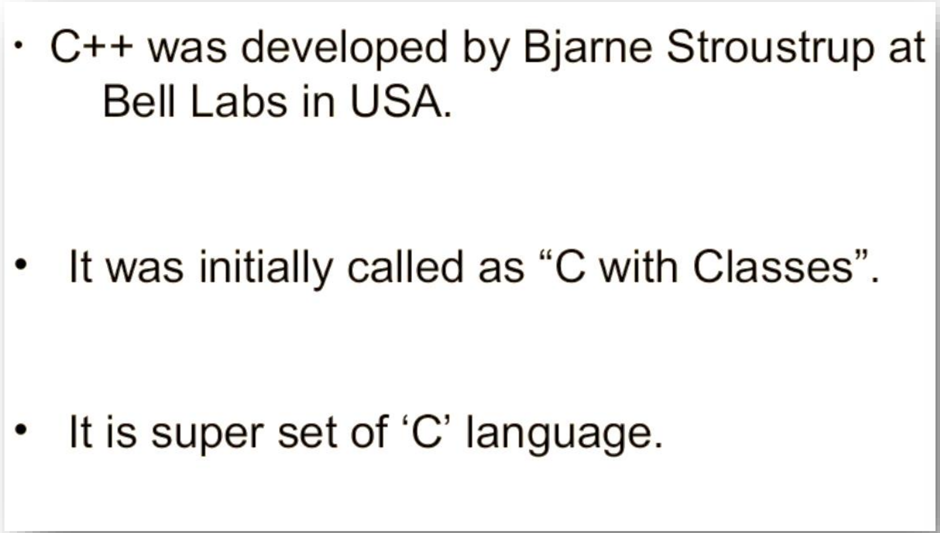
Programming

الذی علم بالقلم۔ علم الانسان ما لم یعلم۔

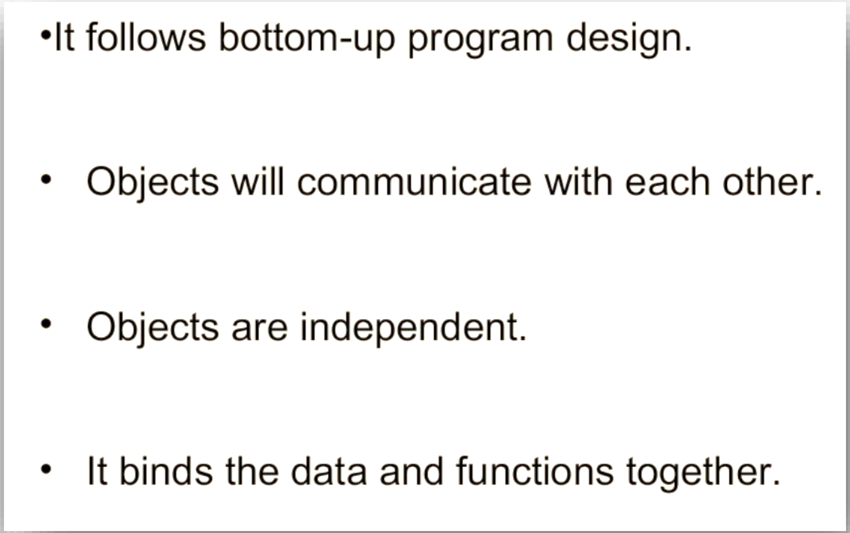
Contents

Repeat Previous Lab

History of C++



- C++ was developed by Bjarne Stroustrup at Bell Labs in USA.
- It was initially called as “C with Classes”.
- It is super set of ‘C’ language.



•It follows bottom-up program design.

- Objects will communicate with each other.
- Objects are independent.
- It binds the data and functions together.

Procedure Oriented Programming (POP)

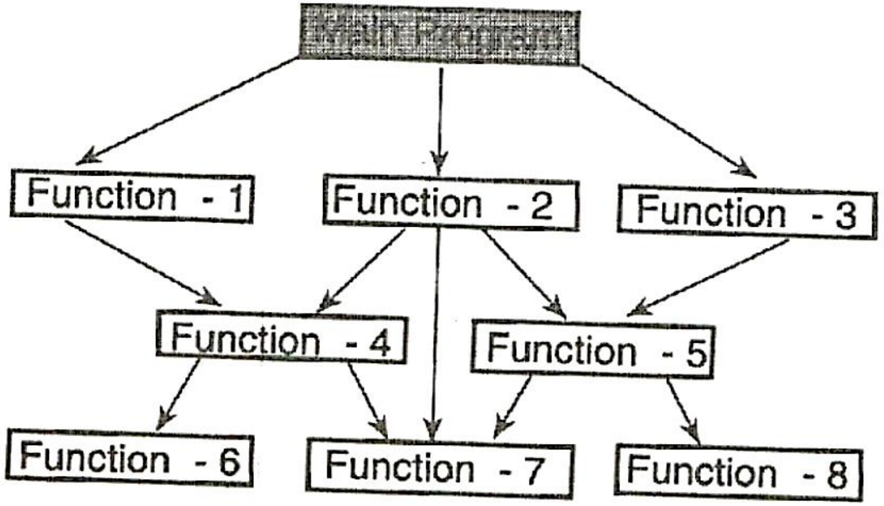
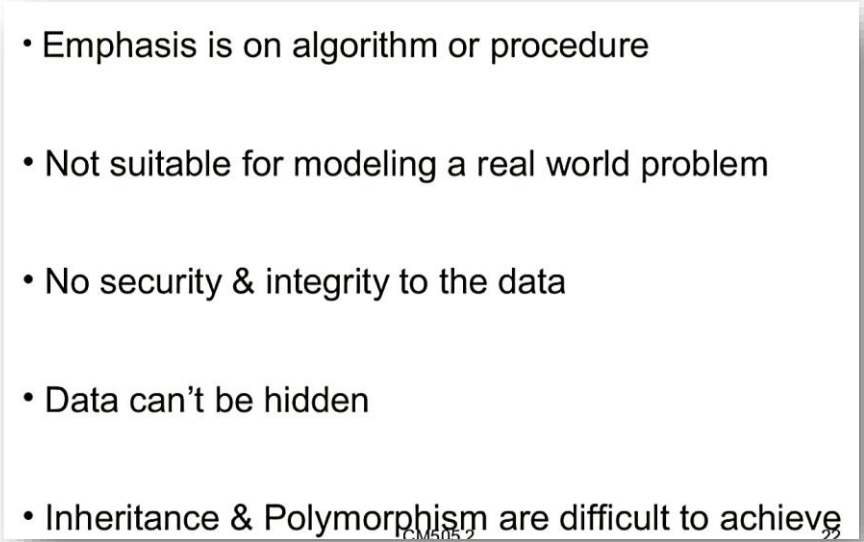


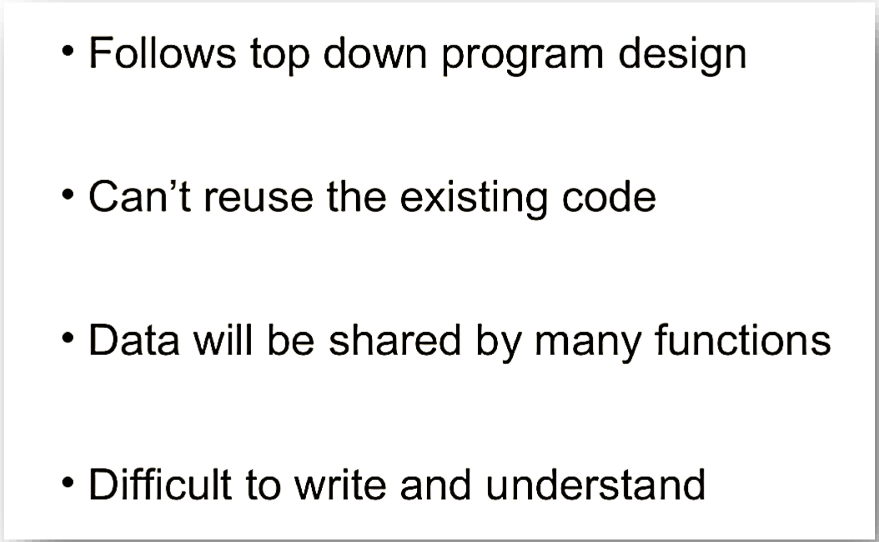
FIG 2.1 *Typical structure of procedure-oriented programs*

Limitation of POP



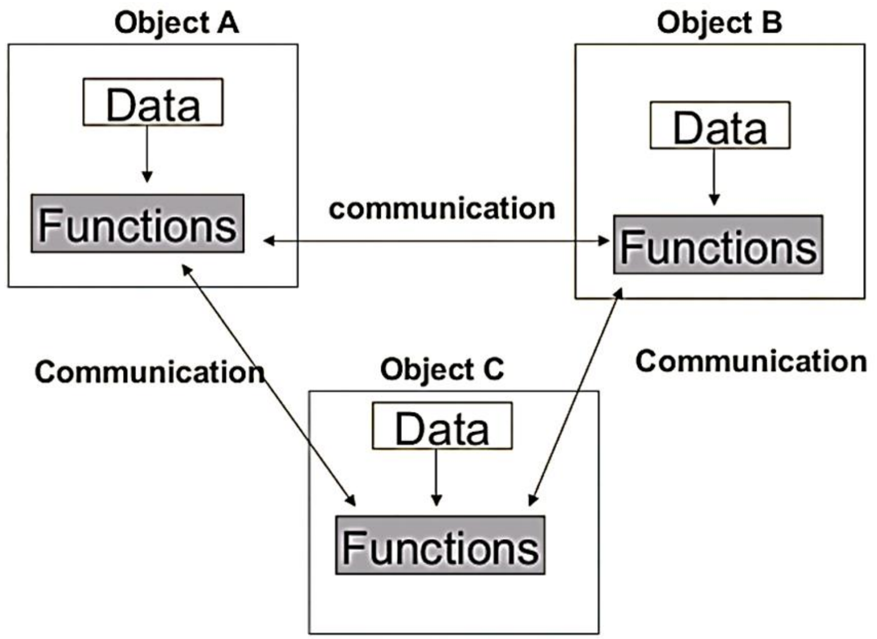
- Emphasis is on algorithm or procedure
- Not suitable for modeling a real world problem
- No security & integrity to the data
- Data can't be hidden
- Inheritance & Polymorphism are difficult to achieve

CM505.2 22



- Follows top down program design
- Can't reuse the existing code
- Data will be shared by many functions
- Difficult to write and understand

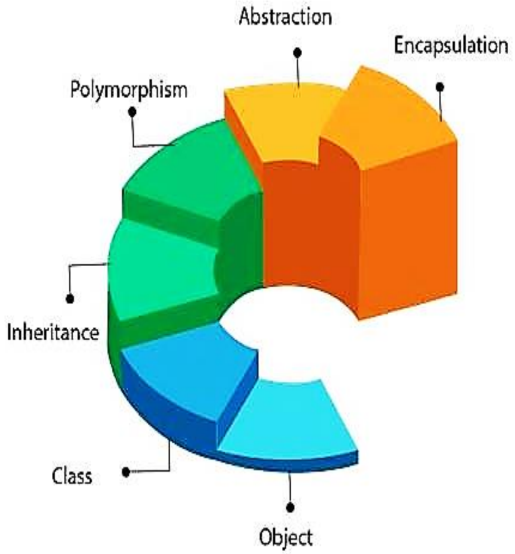
Organization Of Data & Function In OOP



Object Oriented Programming (OOP)

- The object-oriented paradigm is a programming methodology that promotes the efficient design and development of software systems using reusable components that can be quickly and safely assembled into larger systems.

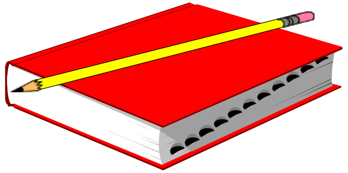
- The main aim of object-oriented programming is to implement real- world concepts like
- Object □ real world entity
- Classes □ Templates/ Blueprints
- Abstraction □ Visibility Controls
- Inheritance □ Backward Compapatibilty , parent child relation
- Polymorphism □Many forms

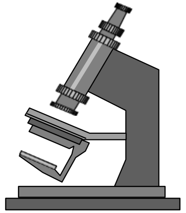
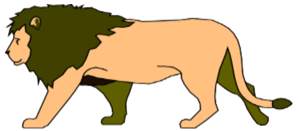


Object Oriented Programming (OOP)

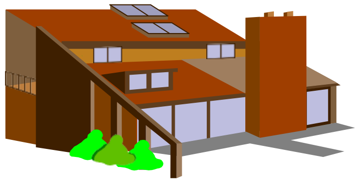
OOP tries to model the real world.
What does the real world look like?

Objects everywhere...

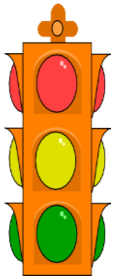


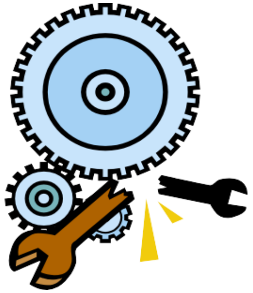
Real world entities



Objects have state...

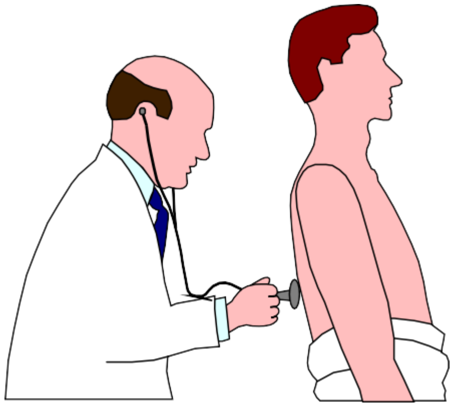


Red



Broken

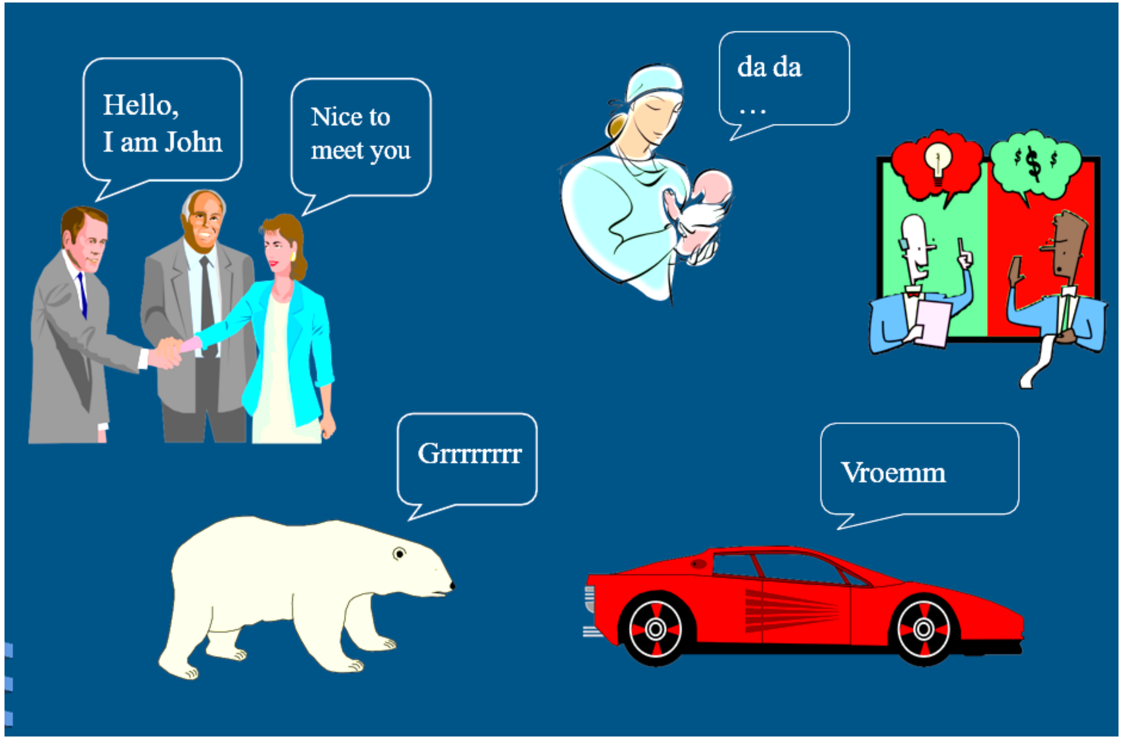
Lying



ill

Happy

Objects have behaviour….



Real World

- The world is a set of things interacting with each other.
- OOP is more natural to humans, but less natural to computers
- Computers (usually) have a single thread of control, so objects take turns

Describing the world

- **Describe a particular person**
- A man has long blond hair, green eyes, is 1.63m tall, weighs 56Kg and studies computer engineering. Now lying down asleep.

- Mahmud studies electronics, has short black hair and brown eyes.
- He is 180cm and 75 kilos. Now running to class!

- **Notice how all have specific values of**
- name, height, weight, eye color, state, …

Features Of Object Oriented Programming

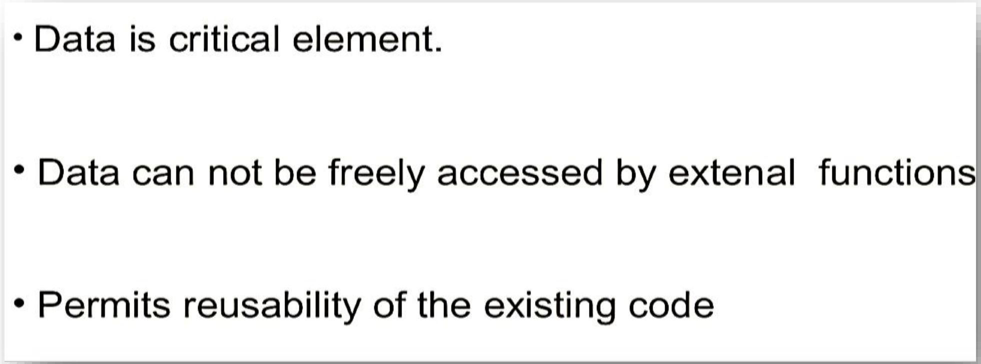
- Emphasis is on data rather than the procedure
- Both data and functions are combined into a single unit
- Data can't be accesssed directly

- Data is hidden and can't be accessed by external functions
- Programs are divided into elements known as objects
- Objects may communicate with each other through functions

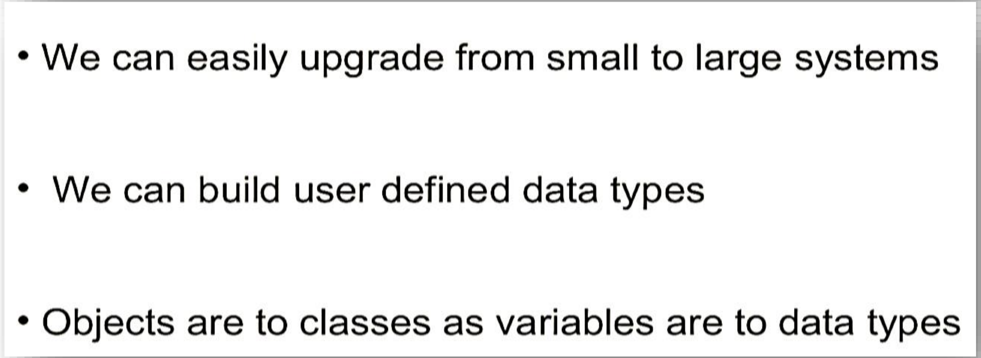
- Higher productivity
- Provides multiple instances of an object
- Provides security to the data
- Easy to write and understand a program

- New data and functions can be easily added
- Follows bottom up approach of program design
- We can eliminate the redundant code
- Time will be saved

Features Of Object Oriented Programming



- Data is critical element.
- Data can not be freely accessed by extenal functions
- Permits reusability of the existing code



- We can easily upgrade from small to large systems
- We can build user defined data types
- Objects are to classes as variables are to data types

Features Of Object Oriented Programming

- Artificial Intelligence & Expert systems
- Simulation & modeling
- OO databases
- Hypertext, Hyper media and Expertext
- CAD / CAM / CAE

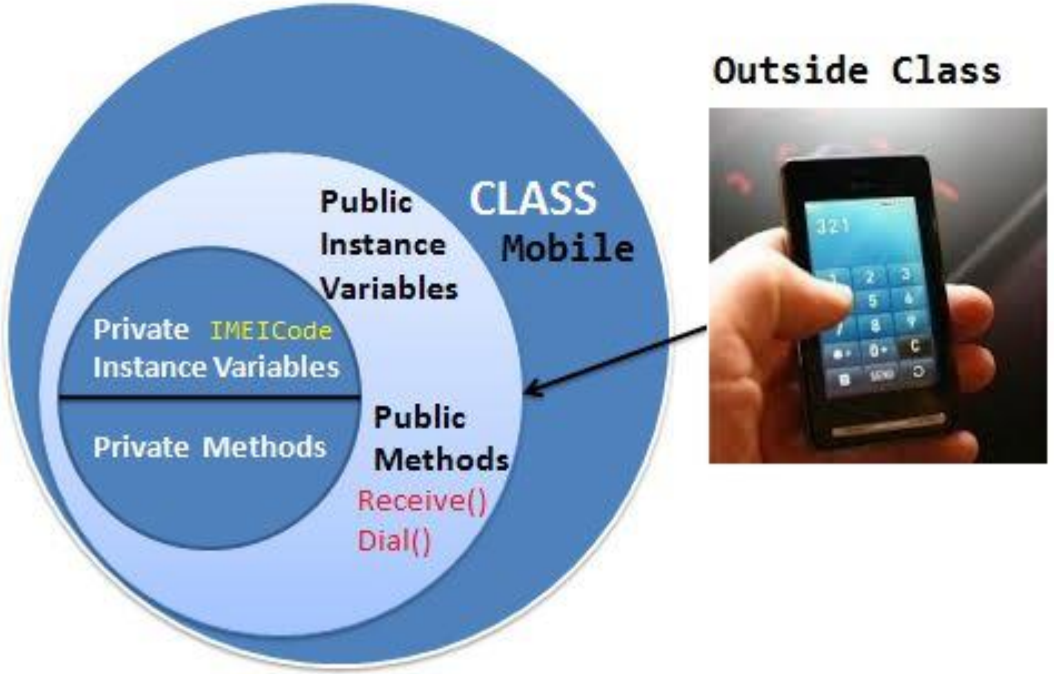
- Decision support system
- Neural Networks
- Real time systems
- Multimedia applications
- GUI, CBTs, Office automation etc

Class

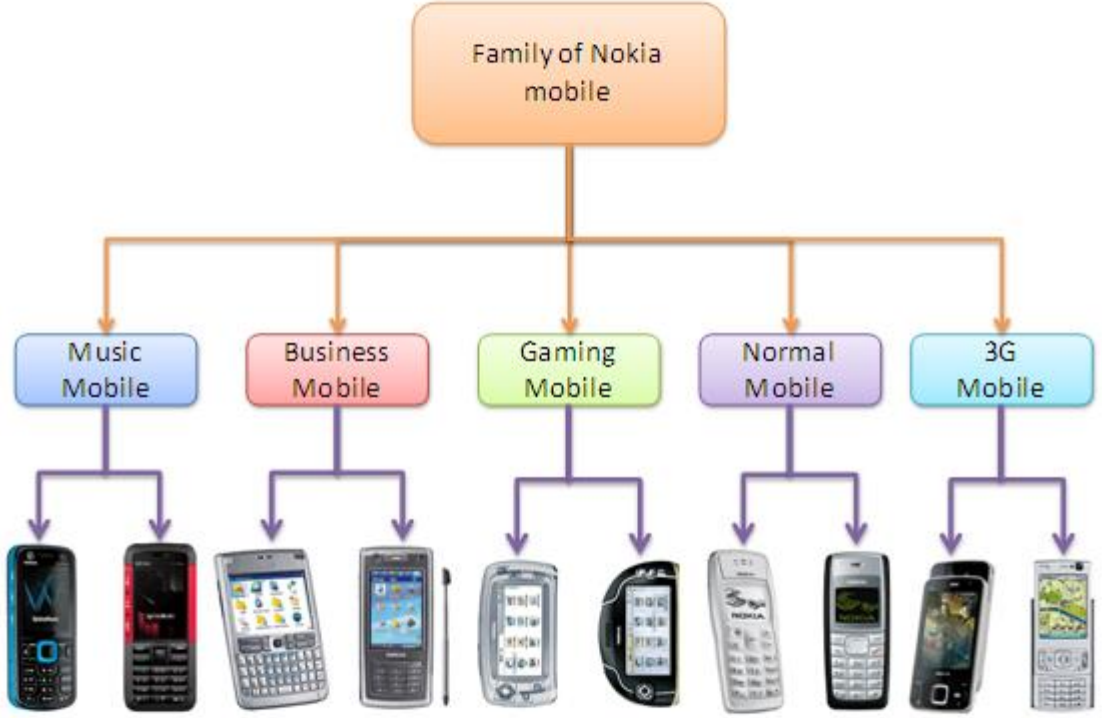
- A Class is a collection of data and functions. The data items and functions are defined within the class. Functions are written to work upon the data items and each function has a unique relationship with data items of the class.
- Classes are defined to create user defined data types. These are similar to built in data types available in all programming languages.
- Definition of data type does not create any space in the computer memory. When a variable of that data type is declared, a memory space is reserved for that variable. Similarly, when a class is defined, it does not occupy any space in the computer memory. It only defines the data items and the member function that can be used to work upon its data items. Thus defining a class only specifies its data members and the relationship between the data items through it functions.

Lab 10 Started…

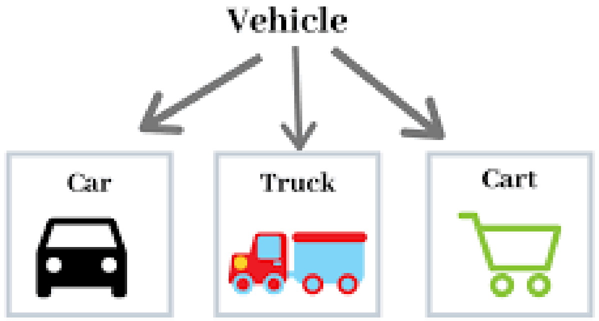
Class

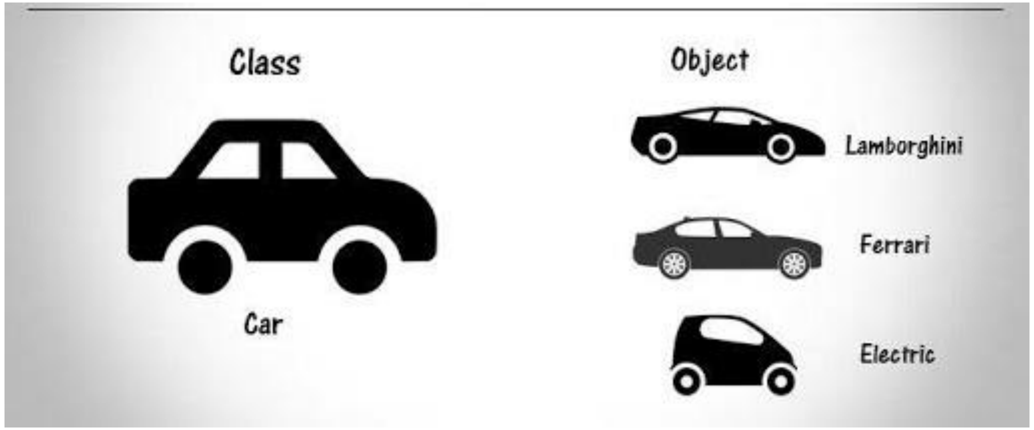

CLASS
Mobile
Public
Instance
Variables
Private IMEICode
Instance Variables
Private Methods
Public
Methods
Receive()
Dial()
Outside Class

Class

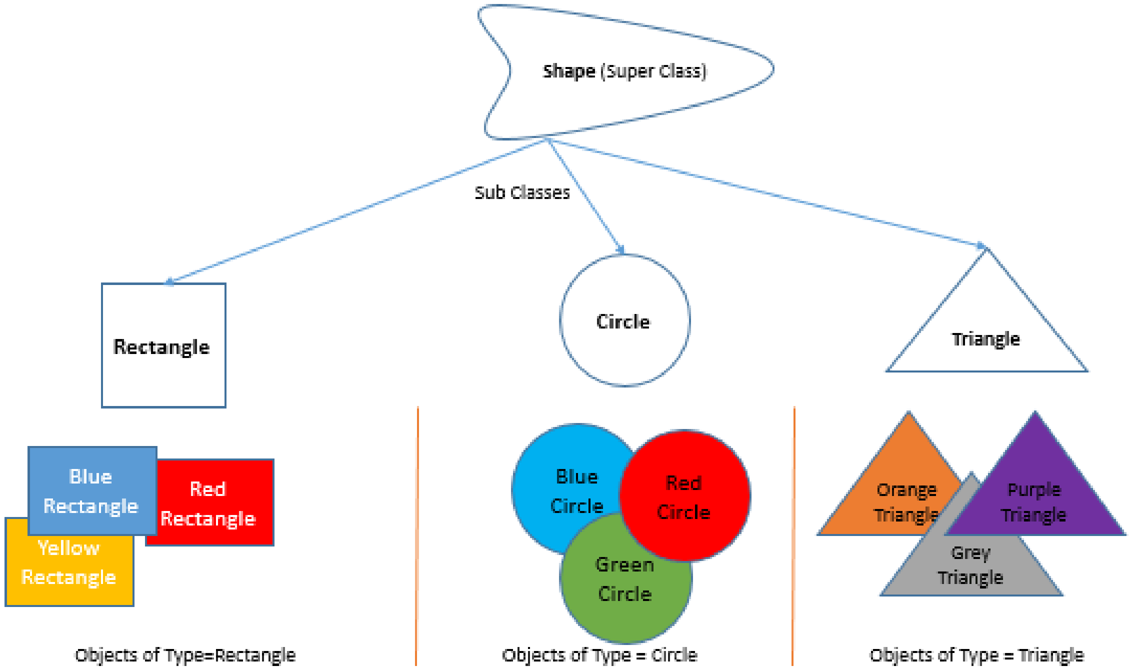

Family of Nokia mobile
Music Mobile
Business Mobile
Gaming Mobile
Normal Mobile
3G Mobile

Class


Vehicle
Car
Truck
Cart

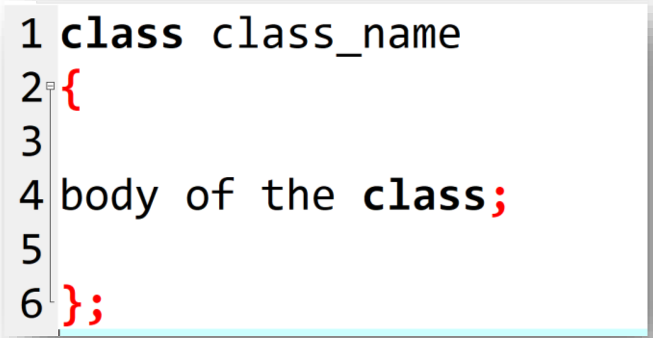

Class
Object
Car
Lamborghini
Ferrari
Electric

Class


Shape (Super Class)
Sub Classes
Rectangle
Circle
Triangle
Blue Rectangle
Red Rectangle
Yellow Rectangle
Blue Circle
Red Circle
Green Circle
Orange Triangle
Purple Triangle
Grey Triangle
Objects of Type=Rectangle
Objects of Type = Circle
Objects of Type = Triangle

Defining a class

A class is defined in a similar way as structure is defined. The keyword “**class**” is used to define the class. The general syntax to define a class is:



```
class class_name
{

body of the class;

};
```

class is a keyword that is used to define a class.

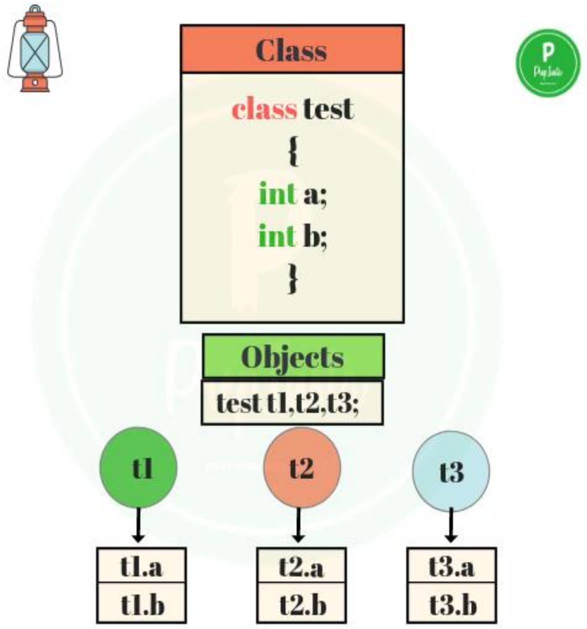
class_name It represents the name of the class.

body of classs The body of the class consist of the data items and the functions. These are called members of the class. These are written between braces.

Semicolon (;) The body of a class ends with semicolon.

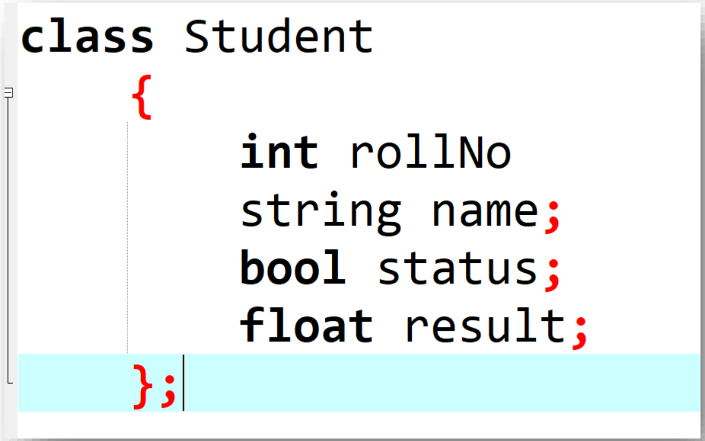
Members of a class

- A class contains data items and functions. These are called members of the class.
- The data items are called **data members** and the functions are called **member functions**.



Data Members

- The data items of a class are called data members of the class. For example a class that has four integer type and two float type data items is declared as:



```
class Student
    {
        int rollNo
        string name;
        bool status;
        float result;
    };
```

In this class **int rollNo**

string name;

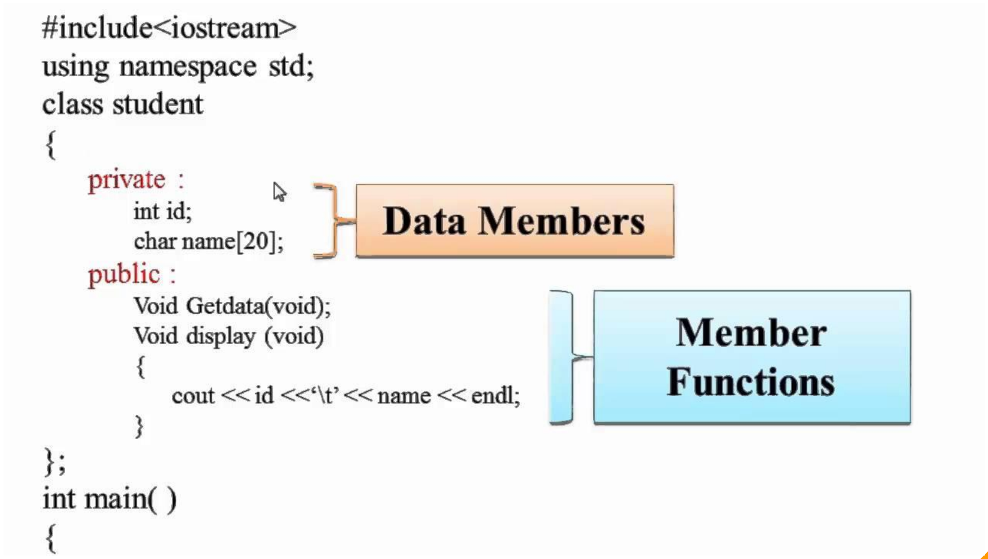
bool status;

float result;

are data members of the class “**Student**”.

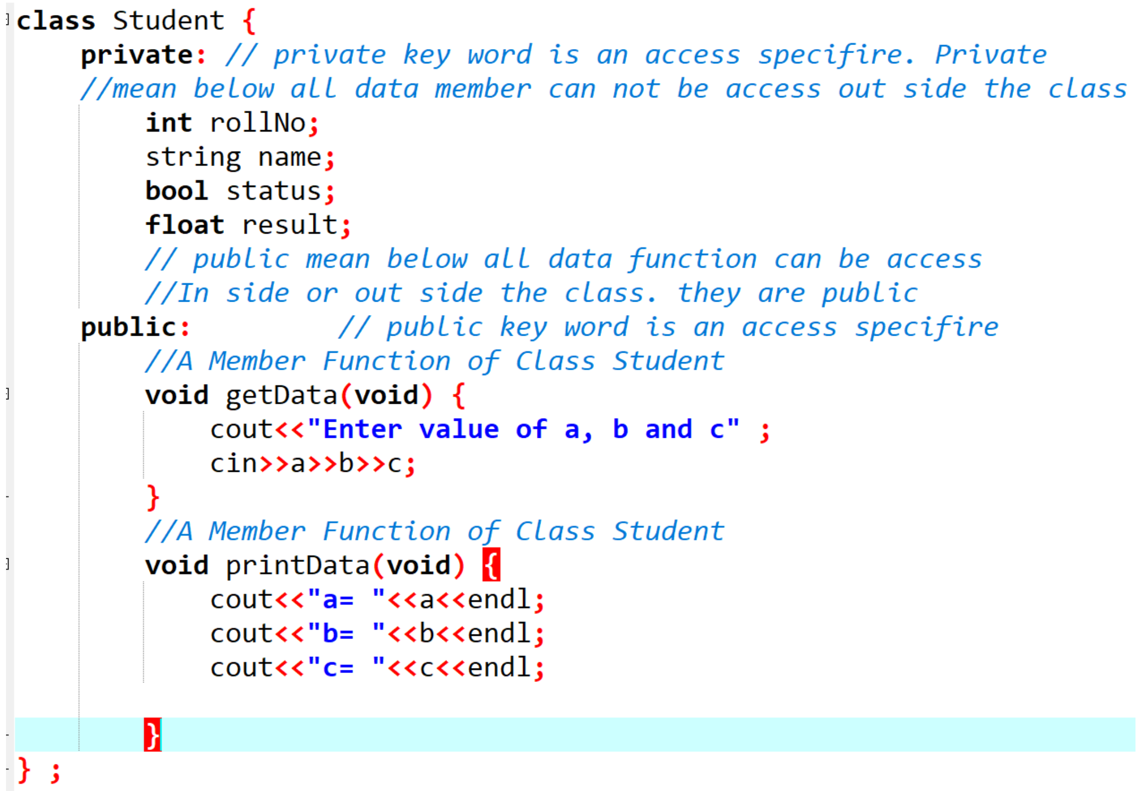
Member Functions

- The functions of a class that are defined to work on its data members are called member functions of the class. The member functions may be defined within the class or outside it. For example:



```
#include<iostream>
using namespace std;
class student
{
    private :
        int id;
        char name[20];
    public :
        Void Getdata(void);
        Void display (void)
        {
            cout << id <<'\t' << name << endl;
        }
};
int main( )
{
```

Member Functions …



```
class Student {
    private: // private key word is an access specifire. Private
    //mean below all data member can not be access out side the class
        int rollNo;
        string name;
        bool status;
        float result;
        // public mean below all data function can be access
        //In side or out side the class. they are public
    public:          // public key word is an access specifire
        //A Member Function of Class Student
        void getData(void) {
            cout<<"Enter value of a, b and c" ;
            cin>>a>>b>>c;
        }
        //A Member Function of Class Student
        void printData(void) {
            cout<<"a= "<<a<<endl;
            cout<<"b= "<<b<<endl;
            cout<<"c= "<<c<<endl;

        }
} ;
```

In this class, there are three data members and two member functions. The member functions are "**getData**" and "**printData**". The "**getData**" function is used to input values into data members **a, b** and **c**. The "**printData**" function is used to print values of the data members on the computer screen.

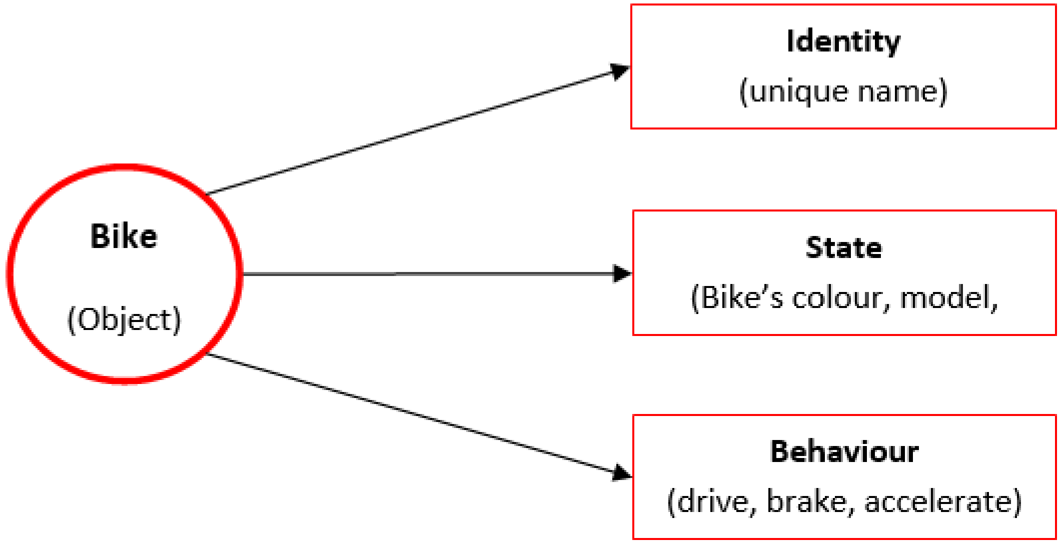
Objects

- A data type is used to declare a variable. A variable of a data type is also known as the instance or case of that data type.
- Each variable has unique name but each variable follows the rules of its data type. When a variable of a data type is declared, some space is reserved for it in the memory.
- A class is also like a data type. It is therefore used to declare variables or instances. The variables or instances of a class are called **objects**.
- A class may contain several data items and functions. Thus the object of a class consists of both the data members and member functions of the class. The combining of both the data and the functions into one unit is called data **encapsulation**.

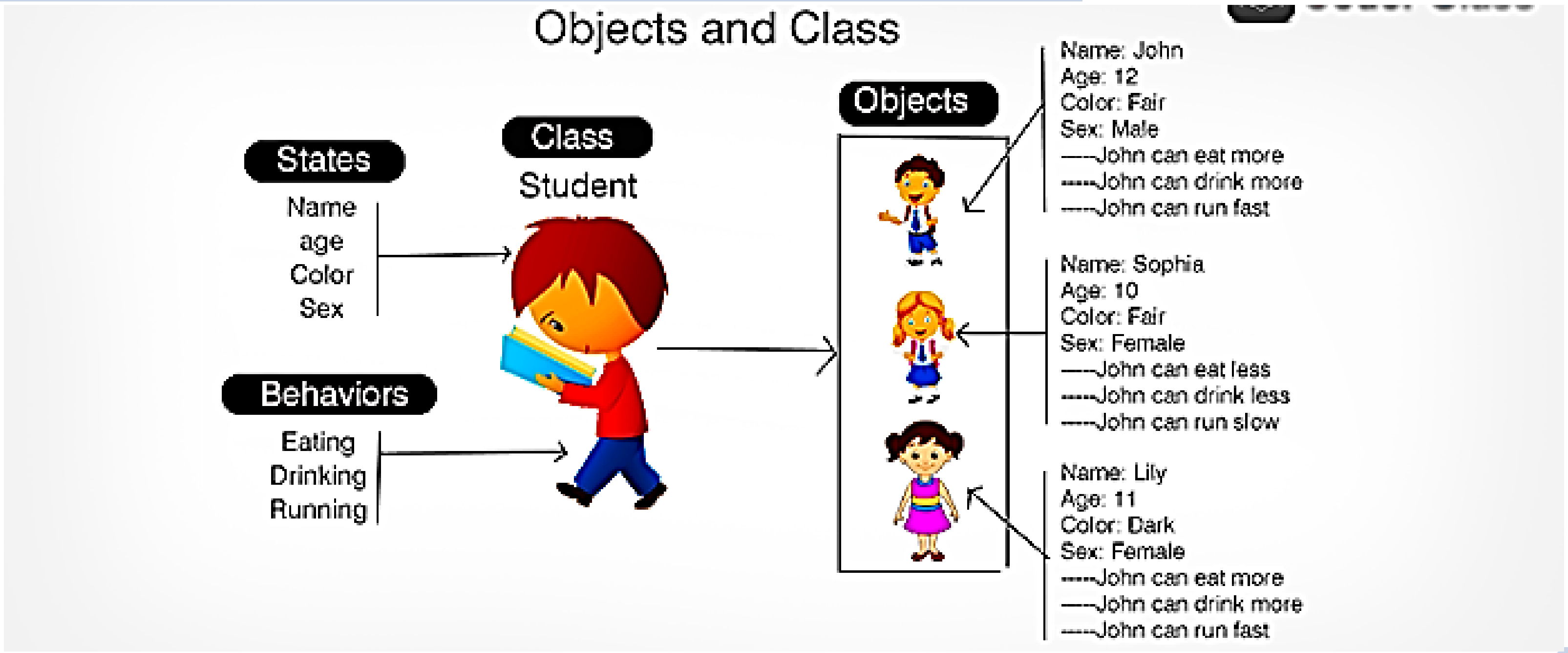
Objects…

- An object represents data members of a class in the memory. Each object of class has unique name. The name of an object differentiates it from other objects of the same class. The values of data members of different objects may be different or same. The values of data members in an object are known as the **state** of the object.
- The functions in an object are called the member functions. They are also known as the them methods. The member functions are used to process and access data of the objects.

Characteristics of Object (Identity, State & Behavior)



Characteristics Object (Identity, State & Behaviour)



Declaring object of a class

- The objects of a class are declared in the similar way as the variables of any data or structure type are declared.
- When a class is defined, no space is reserved for it in the memory. It only provides information how its object will look like. When an object of a class is declared, a memory space is reserved for that object.
- The syntax to create objects of a class type is:

  class_name object_name separated by space;

- For example, to define an objects of Student class, the statement is written as:

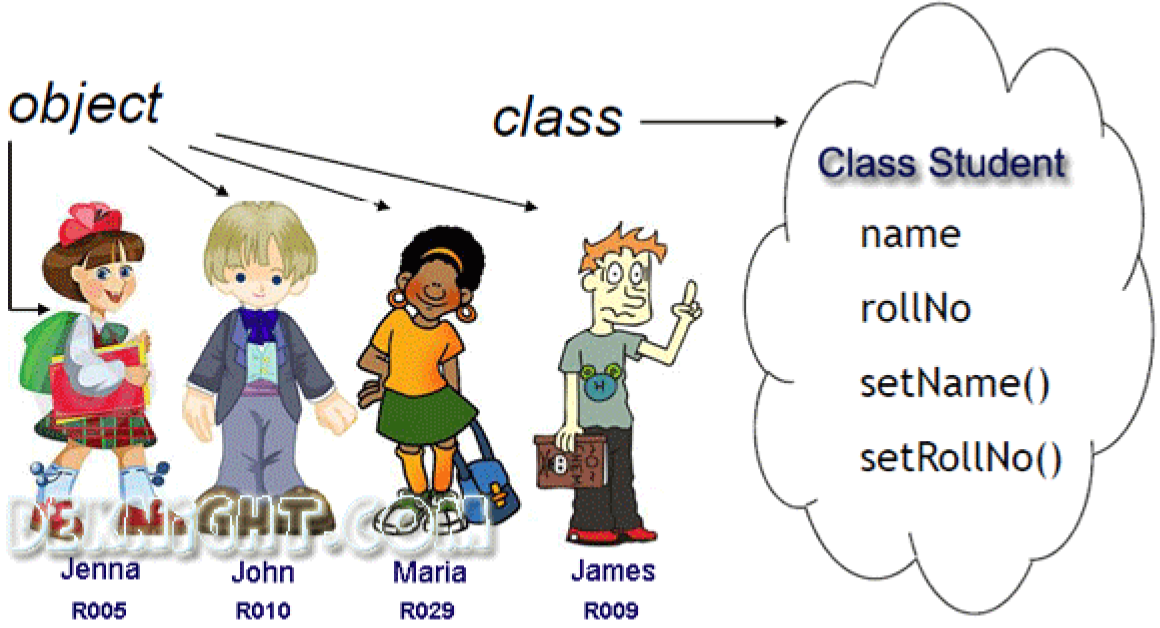
  Student s1;

  Student s2;

Declaring object of a class

- In the above statement two objects namely **s1 ,s2**is declared of Students class. It is the declaration of an object that actually creates an object in the memory.

```
Student std1, std2, std3, std4;

Std1.name= “Jeena”;

Std2.name= “John”;

Std3.name= “Maria”;

Std4.name= “James”;
```



A Class is a Full-Fledged Type

- A class is a type just like **int** and double. You can have variables of a class type, you can have parameter of class type, a function can return a value of a class type, and more generally, you can use a class type like any other type.
- **For example:**
- Student s1,s2,s3;
- Int x,y,z;
- String name, email;

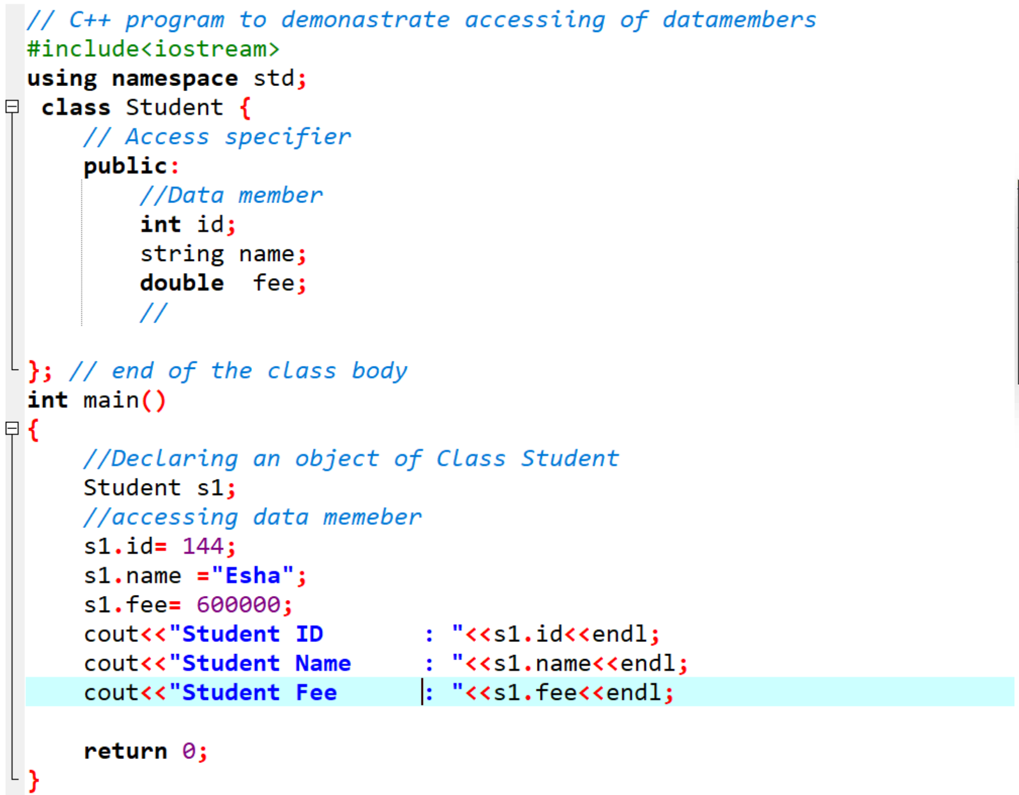
Accessing data members and member functions

- The data members and member functions of class can be accessed using the dot('.') operator with the object.
- For example if the name of object is *obj* and you want to access the member function with the name *printName()* then you will have to write *obj.printName()* .

Accessing Data Members

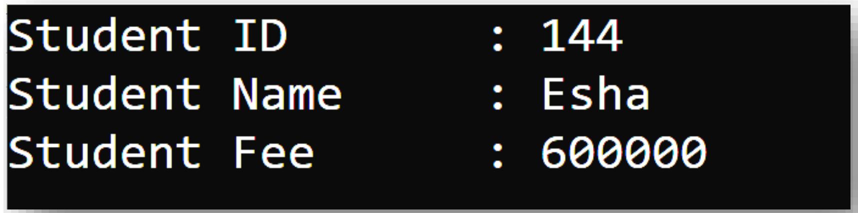
- The public data members are also accessed in the same way given however the private data members are not allowed to be accessed directly by the object.
- Accessing a data member depends solely on the access control of that data member. This access control is given by Access modifiers in C++.
- There are three access modifiers : **public, private and protected**.

Accessing Data Members…



```cpp
// C++ program to demonastrate accessiing of datamembers
#include<iostream>
using namespace std;
 class Student {
    // Access specifier
    public:
        //Data member
        int id;
        string name;
        double  fee;
        //

}; // end of the class body
int main()
{
    //Declaring an object of Class Student
    Student s1;
    //accessing data memeber
    s1.id= 144;
    s1.name ="Esha";
    s1.fee= 600000;
    cout<<"Student ID       : "<<s1.id<<endl;
    cout<<"Student Name     : "<<s1.name<<endl;
    cout<<"Student Fee      : "<<s1.fee<<endl;

    return 0;
}
```

Output:



```
Student ID          : 144
Student Name        : Esha
Student Fee         : 600000
```

Accessing Member Functions

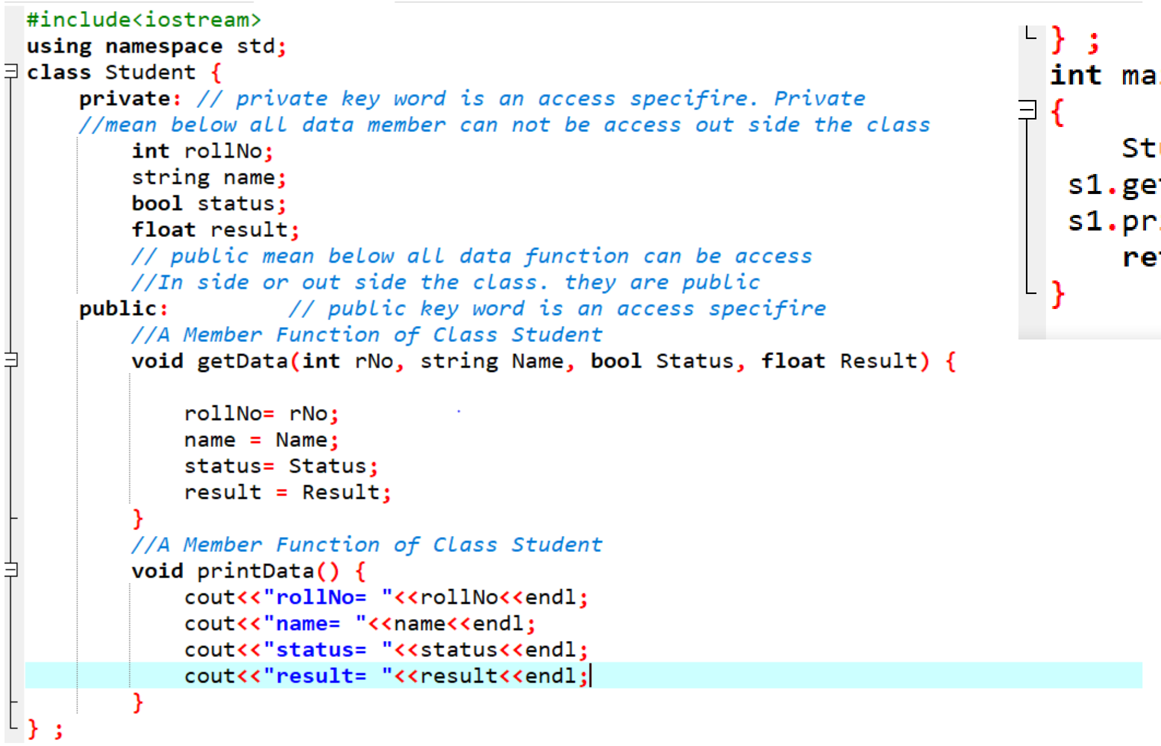
- The member functions of a class is called or accessed in similar way as member or data item of a structure is called.
- The member function is called/accessed through an object of the class.
- The dot operator is used. The dot operator connects the **object name** and **member function**.
- For example, if “**add**” is the name of the object and “**pdate()**” is the member function then the member function is called as shown below:

```
add.pdate();
```

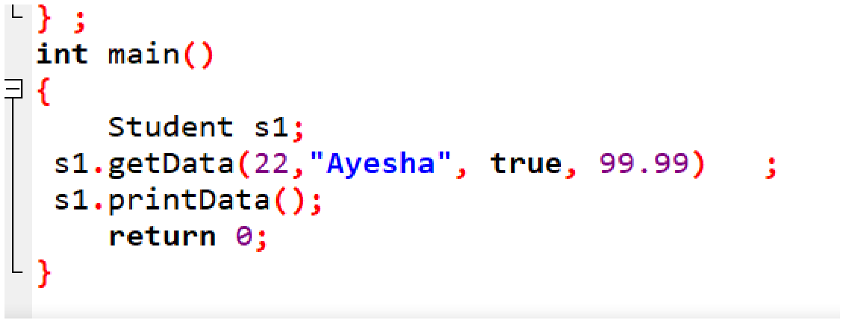
Accessing Member Functions…

- The **dot operator** is also called the class member **access operator**.
- Only those member functions can be accessed from outside the class with dot operator that have been declared as public.

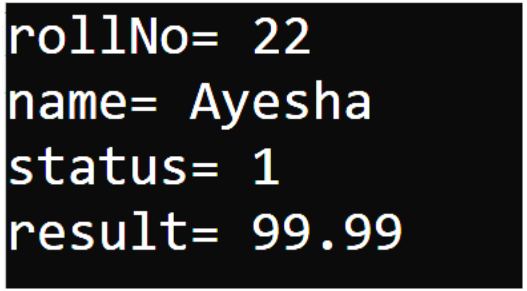
Accessing Member Functions…



```
#include<iostream>
using namespace std;
class Student {
    private: // private key word is an access specifire. Private
    //mean below all data member can not be access out side the class
        int rollNo;
        string name;
        bool status;
        float result;
        // public mean below all data function can be access
        //In side or out side the class. they are public
    public:         // public key word is an access specifire
        //A Member Function of Class Student
        void getData(int rNo, string Name, bool Status, float Result) {

            rollNo= rNo;
            name = Name;
            status= Status;
            result = Result;
        }
        //A Member Function of Class Student
        void printData() {
            cout<<"rollNo= "<<rollNo<<endl;
            cout<<"name= "<<name<<endl;
            cout<<"status= "<<status<<endl;
            cout<<"result= "<<result<<endl;
        }
} ;
```



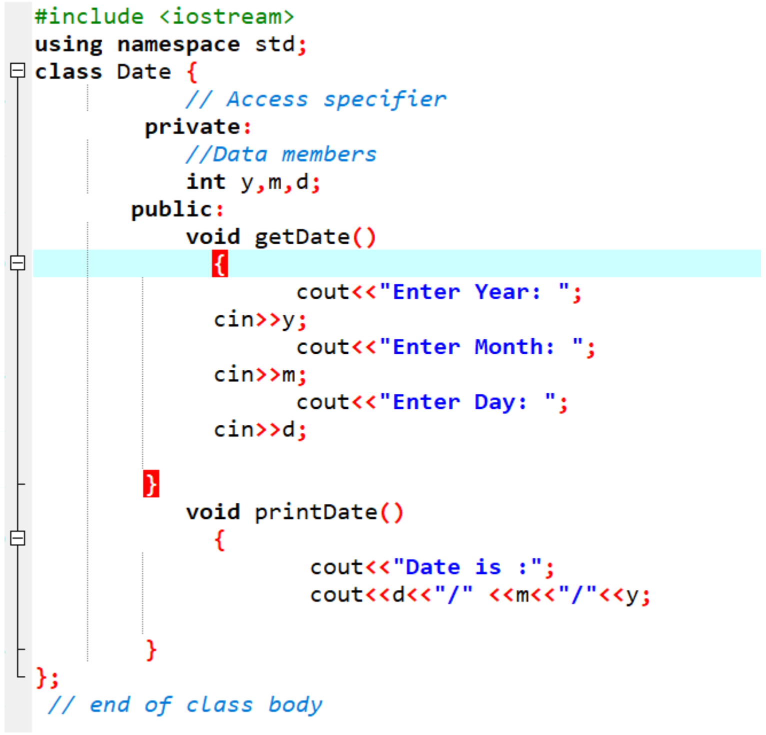
```
} ;
int main()
{
     Student s1;
 s1.getData(22,"Ayesha", true, 99.99)    ;
 s1.printData();
     return 0;
}
```



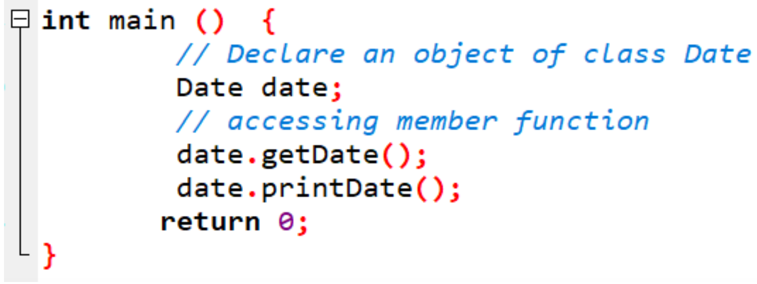
```
rollNo= 22
name= Ayesha
status= 1
result= 99.99
```

Accessing Member Functions…

Program 01 : Write a program to input a date and print on the screen using class.



```cpp
#include <iostream>
using namespace std;
class Date {
            // Access specifier
        private:
            //Data members
            int y,m,d;
       public:
            void getDate()
              {
                    cout<<"Enter Year: ";
              cin>>y;
                    cout<<"Enter Month: ";
              cin>>m;
                    cout<<"Enter Day: ";
              cin>>d;

          }
            void printDate()
              {
                     cout<<"Date is :";
                     cout<<d<<"/" <<m<<"/"<<y;

          }
};
 // end of class body
```



```cpp
int main ()  {
          // Declare an object of class Date
          Date date;
          // accessing member function
          date.getDate();
          date.printDate();
         return 0;
}
```

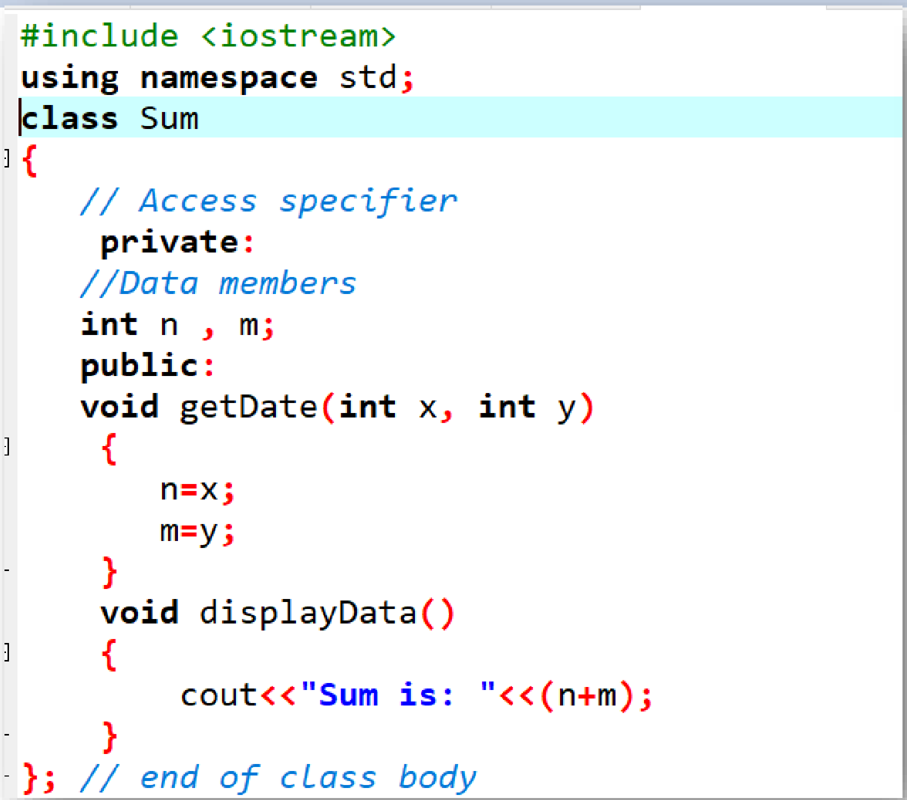
Output:

```
Enter Year: 1995
Enter Month: 12
Enter Day: 12
Date is :12/12/1995
```

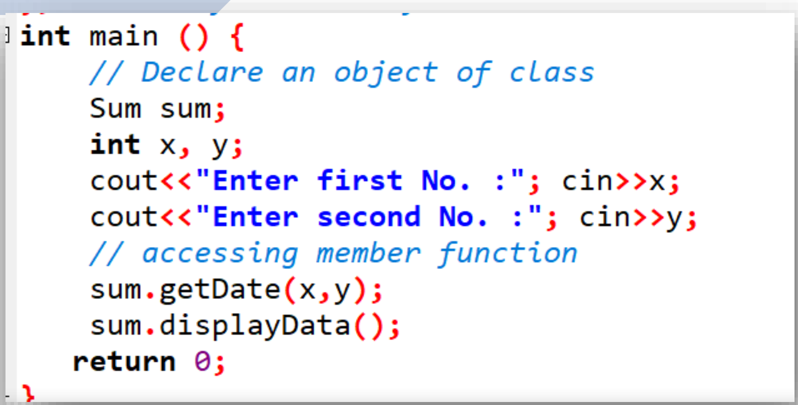
Accessing Member Functions…

Program 02: Write a program by using class to input values using a member functions of a class. Display the sum of two values by using another member function of the class.

Accessing Member Functions…



```
#include <iostream>
using namespace std;
class Sum
{
   // Access specifier
    private:
   //Data members
   int n , m;
   public:
   void getDate(int x, int y)
    {
        n=x;
        m=y;
    }
    void displayData()
    {
         cout<<"Sum is: "<<(n+m);
    }
}; // end of class body
```



```
int main () {
    // Declare an object of class
    Sum sum;
    int x, y;
    cout<<"Enter first No. :"; cin>>x;
    cout<<"Enter second No. :"; cin>>y;
    // accessing member function
    sum.getDate(x,y);
    sum.displayData();
   return 0;
}
```

Output:

```
Enter first No. :4
Enter second No. :4
Sum is: 8
```

Accessing Member Functions…

Program 03: Write a program to input the name of student and marks of three subjects, calculate the total marks and average marks. Each subjects has maximum of 100 marks.

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

class StudentRecord
{
   private:
    char name[15];
    float s1,s2,s3, total, avg;
```

Accessing Member Functions…

```
public:
    void getRecord()
    {
       cout<<"Enter Name of the student: "; cin>>name;
       cout<<"Enter marks of 1st subject: "; cin>>s1;
       cout<<"Enter marks of 2nd subject: "; cin>>s2;
       cout<<"Enter marks of 3rd subject: "; cin>>s3;
       total= s1+s2+s3;
       avg= total/3.0;
    }
```

Accessing Member Functions…

```
 void displayRecord()
    {
       cout<<"Name of the student : "<<name<<endl;
       cout<<"Marks of 1st subject : "<<s1<<endl;
       cout<<"Marks of 2nd subject : "<<s2<<endl;
       cout<<"Marks of 3rd subject : "<<s3<<endl;
       cout<<"Total Marks : "<<total<<endl;
       cout<<"Average Marks : "<<avg<<endl;
    }
}; // end of class body
```

Accessing Member Functions…

```
int main () {

    // Declare an object of class
    StudentRecord stdRecord;

    // accessing member function
    stdRecord.getRecord();
    stdRecord.displayRecord();
    return 0;
}
```

Output:

```
Enter Name of the student: Aousaf
Enter marks of 1st subject: 55
Enter marks of 2nd subject: 77
Enter marks of 3rd subject: 88

Name of the student : Aousaf
Marks of 1st subject : 55
Marks of 2nd subject : 77
Marks of 3rd subject : 88
Total Marks : 220
Average Marks : 73.3333
```

Accessing Member Functions…

```
Enter Name of the student: Yousaf
Enter marks of 1st subject: 55
Enter marks of 2nd subject: 77
Enter marks of 3rd subject: 88
Name of the student : Yousaf
Marks of 1st subject : 55
Marks of 2nd subject : 77
Marks of 3rd subject : 88
Total Marks : 220
Average Marks : 73.3333
```

Accessing Member Functions…

Program 04: Write a program by using class Employee to input the record of employees.

Define the following data members:

- name, bpay, h_rent, ma, gpay

Define the following member functions:

- to input data in name and bpay
- to calculate h_rent, ma, gpay
- to print complete record on the computer Screen.

Where

h-rent = house rent= 60 %

ma = medical allowance   = 20 %

Gpay = bpay + h_rent + ma

Accessing Member Functions…

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

class EmployeeRecord
{
   private:
    char name[15];
    float bpay, h_rent, ma, gpay;

    public:
    void getRecord()
    {
       cout<<"Enter Name of the employee: "; cin>>name;
       cout<<"Enter basic pay of employee: "; cin>>bpay;
    }
```

Accessing Member Functions…

```
void allow()
    {
       h_rent =bpay*60/100;
       ma= bpay*20/100;
       gpay=bpay+h_rent+ma;
    }
    void displayRecord()
    {
       cout<<"Name of the Employee : "<<name<<endl;
       cout<<"Basic Pay            : "<<bpay<<endl;
       cout<<"House Rent           : "<<h_rent<<endl;
       cout<<"Medical Allowance    : "<<ma<<endl;
       cout<<"Net Pay              : "<<gpay<<endl;
    }
}; // end of class body
```

Accessing Member Functions…

```
int main () {

    // Declare an object of class
    EmployeeRecord empRecord;

    // accessing member function
    empRecord.getRecord();
    empRecord.allow();
    empRecord.displayRecord();
    return 0;
}
```

Output:

```
Enter Name of the employee: Aisha
Enter basic pay of employee: 30000

Name of the Employee : Aisha
Basic Pay            : 30000
House Rent           : 18000
Medical Allowance    : 6000
Net Pay              : 54000
```

Defining Member Functions outside class

- Members functions of a class can also be defined outside the class. In this case, only the prototype of the member function is declared inside the class.
- The member functions are defined outside the function in the similar way as user defined functions are defined. However, the scope resolution operator (**::**) is used in the member function declarator to define the function of the class outside the class.
- The general syntax of member function definition outside the class is:

```
type class_name :: function_name (arguments)
{
        body of function
}
```

Defining Member Functions outside class…

- Note: The member function is defined outside its class if the body of the function definition is large. Otherwise the function definition should be defined inside the class.
- To define a member function outside the class definition we have to use the scope resolution :: operator along with class name and function name.

Defining Member Functions outside class…

- **Program 05:** Write a program by using class Employee to input the record of employee by defining the function member outside the class.

```
#include <iostream>
using namespace std;
class EmployeeRecord
{
   private:
    char name[15];
    float bpay, h_rent, ma, gpay;

    public:
     void getRecord(void);
     void allow(void);
     void displayRecord(void);
}; // end of class
```

Defining Member Functions outside class…

```
int main () {

    // Declare an object of class
    EmployeeRecord empRecord;

    // accessing member function
    empRecord.getRecord();
    empRecord.allow();
    empRecord.displayRecord();
    return 0;
}
```

Defining Member Functions outside class…

```
// Definition of getRecord using scope resolution operator ::

   void EmployeeRecord::getRecord()
    {
        cout<<"Enter Name of the employee: "; cin>>name;
        cout<<"Enter basic pay of employee: "; cin>>bpay;
    }
```

Defining Member Functions outside class…

```
// Definition of allow using scope resolution operator ::

  void EmployeeRecord::allow()
   {
       h_rent =bpay*60/100;
       ma= bpay*20/100;
       gpay=bpay+h_rent+ma;
   }
```

Defining Member Functions outside class…

```
// Definition of displayRecord using scope resolution operator ::
    void EmployeeRecord::displayRecord()
    {
        cout<<"Name of the Employee : "<<name<<endl;
        cout<<"Basic Pay            : "<<bpay<<endl;
        cout<<"House Rent           : "<<h_rent<<endl;
        cout<<"Medical Allowance    : "<<ma<<endl;
        cout<<“Net Pay              : "<<gpay<<endl;

    }
```

Defining Member Functions outside class…

Output:

```
Enter Name of the employee: Aisha
Enter basic pay of employee: 30000

Name of the Employee : Aisha
Basic Pay            : 30000
House Rent           : 18000
Medical Allowance    : 6000
Net Pay              : 54000
```

Storage of Object in Memory

- When an of a class is created, a space is reserved in the computer memory to hold its data members. Similarly, separate memory spaces are reserved for each class object.
- The member functions of a class are, however, stored at only one place in the computer memory.
- All objects of the class use the same member functions to process data.
- Therefore while, each object has separate memory space for data members, the member functions of a class are stored in only one place and are shared by all objects of the class.

Storage of Object in Memory…

Program 06:

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

class Temp
{
   private:
    int x;
    float y;

    public:
     void getData(void)
     {
        cout<<"Enter value of x : "; cin>>x;
        cout<<"Enter value of y : "; cin>>y;
     }
```

Storage of Object in Memory…

```
 void print(void)
     {
         cout<<"Entered value of x = "<<x<<endl;
         cout<<"Entered value of y = "<<y<<endl;

     }

}; // end of class body
```

Storage of Object in Memory…

```
int main () {

    // Declare an object of class
    Temp a, b;

    cout<<"Get data data in object a"<<endl;
    a.getData();
    cout<<"Get data data in object b"<<endl;
    b.getData();

    cout<<"Data in object a is : "<<endl;
    a.print();
    cout<<"Data in object b is : "<<endl;
    b.print();

    return 0;
}
```

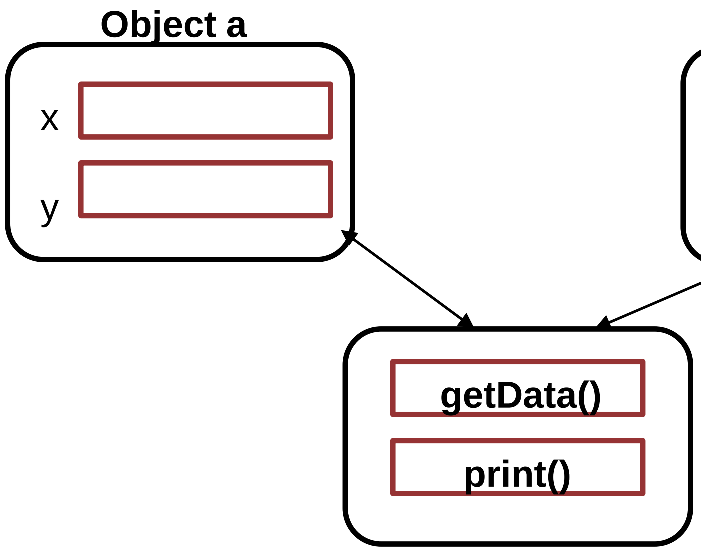
Storage of Object in Memory…

Output:

```
Get data in object a
Enter value of x : 44
Enter value of y : 5

Get data in object b
Enter value of x : 66
Enter value of y : 66

Data in object a is :
Entered value of x = 44
Entered value of y = 5

Data in object b is :
Entered value of x = 66
Entered value of y = 66
```

Storage of Object in Memory…

- The storage of object **a** and **b** as mentioned in the above program example is shown below. These object use the same member functions.



Functions vs Methods

Function — a set of instructions that perform a task.

Method — a set of instructions that are associated with an object.

METHODS

A method, like a function, is a set of instructions that perform a task. The difference is that a method is associated with an object, while a function is not.

References

- https://beginnersbook.com/2017/08/cpp-data-types/
- http://www.cplusplus.com/doc/tutorial/basic_io/
- https://www.w3schools.com/cpp/default.asp
- https://www.javatpoint.com/cpp-tutorial
- https://www.geeksforgeeks.org/object-oriented-programming-in-cpp/?ref=lbp