Some characteristics of Object Oriented Programming (Revisited)

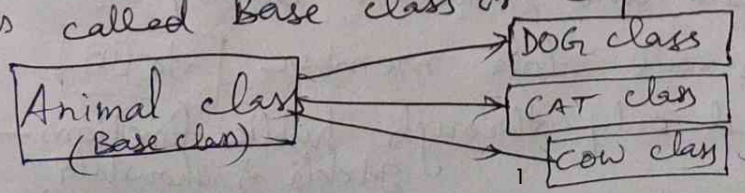
- class
- Object
- Encapsulation
- Inheritance
- Polymorphism
- Abstraction

-Class :- The building block of c++ programming is class. It is user-defined data type which is like a blueprint for an Object.
class has data members and member functions.

Object :- An object is an instance of a class. When a class is defined, no memory is allocated but when it is instantiated (creation of object) then memory is allocated.

Encapsulation :- In general, wrapping up data and functions under a single unit is called encapsulation. In object-oriented programming, Encapsulation is defined as binding together the data and the functions that manipulate them.
Encapsulation also leads to data abstraction or data hiding. Using encapsulation also hides the data.

Inheritance :- The capability of a class to re-use properties and characteristics defined in another class is called Inheritance. It is one of the important feature of Object Oriented programming. The class that inherits the properties from another class is called child class or derived class or sub-class.
The class whose properties are inherited by a sub-class is called Base class or Superclass. Example :-



Let us try to learn this very basic "Hello World" program by using concepts of Object Oriented Programming Concepts:—

```
#include <string.h>
#include <iostream>
using namespace std;
class Hello{
        private:
                char str[15];
        public:
                void getdata(char s[])
                {
                strcpy(str, s);
                }
                void showdata()
                {
                cout << "In Your String:" << str;
                }
};
int main(){
        Hello object1;
        object1.getdata("Hello World");
        object1.showdata();
return 0;
}
```

Explanation of the program:-

First of all we have used strcpy() function which copies the content of `s` to `str`. In above case, "Hello World" is copied to variable `str` from `s`. strcpy() function is defined in string.h file of C programming.

<iostream> is the basic header file for c++.

In main() function we have created object of `Hello` class which is - `object1`.

With the help of object1 we have called both the functions — getdata and showdata one by one.

Class Hello has private data member `str[15]` which can be accessed only through public functions — getdata & showdata.

(10)

To understand Polymorphism, we need to understand class and constructor first of all.

Classes are an expanded concept of data structures — like data structures, they can contain data members, but they can also contain functions as members.

An object is an instantiation of a class. In terms of variables, a class would be the type, and an object would be the variable.

Classes have the same format as plain data structures, except that they can also include functions and have these new things called access specifiers — private, public and protected.

These specifiers modify the access rights for the members that follow them.

private — these members are accessible only from within other members of the same class or from their friend class (also from members of their derived classes)

protected — these members are accessible from other members of the same class (+friend also) but also from members of their derived classes.

public — these are accessible from anywhere where object is visible.

By default, all members of a class have private access for all its members. (If we do not specify any access specifier).

Constructor :- Constructor is a special kind of class member function that is executed automatically when an object of the class is instantiated. It has the same name as the class. Constructors have no return type (not even void).

Constructors are typically used to initialize member variables of the class to appropriate default values, or to allow the user to easily initialize those member variables while creating object.

Now, let us consider following example :-

```
#include <iostream>
using namespace std;
class Hello {
    private:
        string name;
    public:
        Hello() {
            name = "Hello World";
        }
        Hello(string newname)
        {
            name = newname;
        }
        void show() {
            cout << name << endl;
        }
};
int main() {
    Hello person1;
    Hello person2("shershah college");
    person1.show();
    person2.show();
}
```

// Hello World is passed to name

// Shershah College is passed.

// Hello World is printed

// Shershah College is printed

Compile & Run the program you will get :-

```
Hello World
Shershah College
```

In above program, since we have defined two constructors one with no argument and other with string argument. This is also called constructor overloading. Any function with multiple definitions are overloading.