Class 09
Topic: Classes and Objects (C++ OOP Class 02)

```
class Student //template / blueprint
{
// access specifier(Encapsulation) -> 3 -> private , public , protected
// by default private
private:
    // member variable -> attributes -> variables of a class
protected:
    float cgpa;

public:
    char name[100];
    int roll,age;

    //member functions -> methods

    void setter(const char *nm, int r, int a );   // taking input
    void getter(); //displays output
    void fun(int n); //member function declaration
};
```

A **class** is a user-defined data type that acts as a **template or blueprint** for creating objects.

It groups **data members (variables)** and **member functions (methods)** together to represent an entity.

Access specifiers control visibility:

Private – only accessible within the class (default).

Protected – accessible within the class and derived classes.

Public – accessible from anywhere using an object.

```
void Student :: setter(const char *nm, int r, int a )
{
    cout<<"For input"<<endl;
    strcpy(name, nm);
    roll = r;
    age = a;
}
```

The `setter()` function is used for **taking input** and assigning values to data members.

`strcpy()` (from `<cstring>`) copies a string into the character array `name`.

This shows **encapsulation**, as we access and modify private/protected data using public methods.

```
void Student :: getter()
{
    cout<<"For output"<<endl;
    cout<<"Name : "<<name<<endl;
    cout<<"Roll : "<<roll<<endl;
    cout<<"Age : "<<age<<endl;
}
```

The `getter()` function is used for **displaying output**.

It prints all data stored in the object using `setter()`.

This demonstrates **data access through public functions**.

```
void Student :: fun(int n) //function definition outside class

{

    int x=90;

    cout<<x;

    cout<<name<<roll<<age<<endl;

    cout<<cgpa<<endl;

}
```

The `fun()` function is defined **outside the class** using the **scope resolution operator (::)**.

It shows how to access both public and protected members from inside a class method.

`x` is a local variable, while `cgpa` is a protected member, hence accessible only inside the class.

```
int main()

{

    // objects -> instance(real world usable copy) of a class

    Student ob1, ob2;  // ob1 is an object of Student class


    cout<<endl;


    ob1.setter("Tahsin", 2310034, 21);

    cout<<endl;


    ob1.getter();

    cout<<endl;
    cout<<endl;


    ob2.setter("Maria",1087,20);
    ob2.getter();


    ob2.fun(2);
}
```

Objects are **instances of a class** — real, usable copies of the template.

Each object has its **own memory** for all data members.

`setter()` and `getter()` are called using the object name and dot operator.

Protected members like `cgpa` cannot be accessed directly in `main()`.