### Accessing Data Members of Class

Accessing a data member depends solely on the access control of that data member. If its public, then the data member can be easily accessed using the direct member access (.) operator with the object of that class.

If, the data member is defined as private or protected, then we cannot access the data variables directly. Then we will have to create special public member functions to access, use or initialize the private and protected data members. These member functions are also called **Accessors** and **Mutator** methods or **getter** and**setter** functions.

#### Accessing Public Data Members

Following is an example to show you how to initialize and use the public data members using the dot (.) operator and the respective object of class.

class Student

{

public:

int rollno;

string name;

};

int main()

{

Student A;

Student B;

A.rollno=1;

A.name="Adam";

B.rollno=2;

B.name="Bella";

cout <<"Name and Roll no of A is :"<< A.name << A.rollno;

cout <<"Name and Roll no of B is :"<< B.name << B.rollno;

}

#### Accessing Private Data Members

To access, use and initialize the private data member you need to create getter and setter functions, to get and set the value of the data member.

The setter function will set the value passed as argument to the private data member, and the getter function will return the value of the private data member to be used. Both getter and setter function must be defined public.

*Example* :

class Student

{

private: // private data member

int rollno;

public: // public accessor and mutator functions

int getRollno()

{

return rollno;

}

void setRollno(int i)

{

rollno=i;

}

};

int main()

{

Student A;

A.rollono=1; //Compile time error

cout<< A.rollno; //Compile time error

A.setRollno(1); //Rollno initialized to 1

cout<< A.getRollno(); //Output will be 1

}

So this is how we access and use the private data members of any class using the getter and setter methods. We will discuss this in more details later.

#### Accessing Protected Data Members

Protected data members, can be accessed directly using dot (.) operator inside the subclass of the current class, for non-subclass we will have to follow the steps same as to access private data member.