```
//Class to basic conversion is done with the help of operator function
#include<iostream>
using namespace std;
class Time
{
private:
int h,m;
public:
Time()
{
h=m=0;
}
void get_data()
{
cin>>h>>m;
}
operator int()
{
int t=h*60+m;
return t;
}
};
int main()
{
int min;
Time T1;
cout<<"\n Enter the number of hrs and mins";
T1.get_data();
min=T1;//(int)T1(internal representation)
cout<<"\n Total Minutes="<<min;
}
----------------------------------------
//Class to basic conversion is done with the help of operator function
#include<iostream>
using namespace std;
class Distance
{
private:
int km,m;
public:
Distance()
{
km=m=0;
}
void get_data()
{
cin>>km>>m;
}
operator int()
```

```
{
int d=km*1000+m;
return d;
}
};
int main()
{
int metres;
Distance D1;
cout<<"\n Enter the number of kms and metres";
D1.get_data();
metres=D1;
cout<<"\n Total metres="<<metres;
}
```

```
#include<iostream>
using namespace std;
class Circle
{
private:
float radius;
public:
Circle()
{
radius=0.0f;
}
void get_data()
{
cin>>radius;
}
operator float()
{
float t=3.14*radius*radius;
return t;
}
};
int main()
{
float area;
Circle d1;
cout<<"\n Enter the radius of the circle";
d1.get_data();
area=d1;
cout<<"\n area of circle ="<<area;
}
```

```
//Private mode of inheritance(Single level inheritance)
#include<iostream>
using namespace std;
```

```
class A
{
	protected:
		int x,y;
		public:
			void showdataA()
			{
				cout<<x<<" "<<y<<" ";
			}
};
class B:private A
{
	private:
		int z;
		//int x,y;
		public:
			void getdata()
			{
cout<<"\n Enter values of x ,y and z:";
cin>>x>>y>>z;// x and y are private in B, but they are accessible inside class
			}
			void showdataB()
			{
				showdataA();
				cout<<z;
			}
};
int main()
{
	B obj1;
	obj1.getdata();
	obj1.showdataB();
	//obj1.showdataA(); //We cannot access, as showdataA() is private in B
//	  cout<<obj1.x<<" "<<obj1.y;  //We cannot acces, as x and y are private in B
}
----------------------------------------
#include<iostream>
#include<string>
using namespace std;
class student
{
private:
int roll_no;
protected:
char section[10];
public:
void get_rno()
{
cout<<"\n Enter the roll number:";
```

```
cin>>roll_no;
}
void show_rno()
{
cout<<"\n Roll no:"<<roll_no;
}
};
class result:private student
{
private:
float fees;
public:
void get_data()
{
get_rno();
cout<<"\n Enter fees:";
cin>>fees;
cout<<"\n Enter section:";
cin>>section;
}
void display()
{
show_rno();
cout<<"\n Fees:"<<fees;
cout<<"\n Section:"<<section;
}
} ;
int main()
{
result obj1;
obj1.get_data();
obj1.display();
//obj1.get_rno(); //It  not will work-Private data
//obj1.show_rno();//It will not work-Private data
//obj1.roll_no=78;//It will not work(Private data)-Not inherited
//strcpy(obj1.section,"K17MM");//It will not work(Private data)
}
----------------------------------------
#include<iostream>
using namespace std;
class A
{
	protected:
		int x,y;
		public:
			void showdataA()
			{
				cout<<x<<" "<<y<<" ";
			}
```

```
};
class B:protected A
{
	protected:
		int z;
		public:
			void getdata()
			{
cout<<"\n Enter values of x ,y and z:";
cin>>x>>y>>z;// x and y are protected in B also, but they are accessible inside class
			}
			void showdataB()
			{
				showdataA();
				cout<<z;
			}
};
int main()
{
	B obj1;
	obj1.getdata();
	obj1.showdataB();
//	  obj1.showdataA(); //We cannot access, as showdataA() is protected in B
//	  cout<<obj1.x<<" "<<obj1.y;  //We cannot access, as x and y are protected in B
}
---------------------------------------
#include<iostream>
using namespace std;
class A
{
	protected:
		int x,y;
		public:
			void showdataA()
			{
				cout<<x<<" "<<y<<" ";
			}
};
class B:public A
{
	protected:
		int z;
		public:
			void getdata()
			{
cout<<"\n Enter values of x ,y and z:";
cin>>x>>y>>z;// x and y are protected in B also, but they are accessible inside class
			}
			void showdataB()
```

```
                {
                        cout<<z;
                }
};
int main()
{
        B obj1;
        obj1.getdata();
        obj1.showdataA();//showdataA() is public in the derived class B, hence accessible
        obj1.showdataB();
        //cout<<obj1.x<<" "<<obj1.y;  //We cannot access, as x and y are protected in B
}
-------------------------------------
#include<iostream>
#include<string>
using namespace std;
class A
{

protected:
int a;
public:
void inputA(){
cout<<"Enter a";
cin>>a;
}
};
class B:public A
{
private:
int b;
public:
void inputB(){
inputA();
cout<<"Enter b";
cin>>b;
}
int area(){
return a*b;
}
} ;
int main()
{
B b1;
b1.inputB();
cout<<"Area of the rectange is "<<b1.area();
}
```