1. Student:

```
using namespace std;

#include<iostream>

#include<string.h>

struct Student

{

        int roll_no;

        char name[20];

        Student()

        {

                cout<<"\n\ndefault constructor called\n";

                this->roll_no=0;

                strcpy(this->name,"not_given");

        }

        Student(int r,char* n)

        {

                cout<<"\n\nparameterised constructor called\n";

                this->roll_no=r;

                strcpy(this->name,n);

        }

        void setRoll(int r)                    //setters(mumtator)

        {

                this->roll_no=r;

        }

        void setName(const char* n)
        {
                strcpy(this->name,n);     //setters(mumtator)
        }
        int getRoll()                    //getters(accessors)
        {
                return this->roll_no;
        }
        char* getName()           //getters(accessors)
        {
                return this->name;
        }
        void display()
        {
                cout<<"\nroll no "<<this->roll_no<<" is "<<this->name<<"\n";
        }
};
int main()
{
        Student s1;
        int roll_no;
        char name[20];
        s1.display();
        cout<<"\nenter roll no of student: "<<"\n";
        cin>>roll_no;
        cout<<"\nenter name of the student: "<<"\n";
        cin>>name;

        s1.setRoll(roll_no);        //member function

        s1.setName(name);       //member function

        cout<<"\nafter setting value\n";

        s1.display();       //member function

        cout<<"\ngetter\n";

        cout<<"\nRoll no: "<<s1.getRoll() <<"\nName: "<<s1.getName()<<"\n";

        Student s3;

        s3.setRoll(10);

        s3.setName("sachin");

        printf("\nafter setting value\n");

        s3.display();       //member function

        cout<<"\ngetter\n";

        cout<<"\nRoll no: "<<s3.getRoll() <<"\nName: "<<s3.getName()<<"\n";

        Student s2(42,"pragati");           //member function

        s2.display();       //member function

        cout<<"\ngetter\n";

        cout<<"\nRoll no: "<<s2.getRoll()<<"\nname: "<<s2.getName()<<"\n";

        return 0;

}
```

2. Employee:

```
using namespace std;

#include<iostream>

#include<string.h>
```

```
struct Employee
{
	int emp_id;
	char name[20];
	double salary;
	Employee()
	{
		cout<<"\n\ndefault constructor called\n";
		this->emp_id=0;
		strcpy(this->name,"not_given");
		this->salary=0;
	}
	Employee(int i,const char* n,double s)
	{
		cout<<"\n\nparameterised called\n";
		this->emp_id=i;
		strcpy(this->name,n);
		this->salary=s;
	}
	void setId(int i)  //setters(mutators)
	{
		this->emp_id=i;
	}
	void setName(const char* n)      //setters(mutators)
	{
```

```
                strcpy(this->name,n);
        }
        void setSalary(double s) //setters(mutators)
        {
                this->salary=s;
        }
        int getId()        //getters(accessors)
        {
                return this->emp_id;
        }
        char* getName()        //getters(accessors)
        {
                return this->name;
        }
        double getSalary()      //getters(accessors)
        {
                return this->salary;
        }
        void display()
        {
                cout<<"\nemployees detail: \nid: "<<this->emp_id<<"\tname: "<<this-
>name<<"\tsalary: "<<this->salary<<"\n";
        }
};
int main()
{
        Employee e1;

        int emp_id;

        char name[20];

        double salary;

        e1.display();     //member function

        cout<<"\nenter employee id:\n";

        cin>>emp_id;

        cout<<"enter employee name: \n";

        cin>>name;

        cout<<"enter employee salary: \n";

        cin>>salary;

        e1.setId(emp_id);         //member function

        e1.setName(name);      //member function

        e1.setSalary(salary);      //member function

        cout<<"\nafter setting values\n";

        e1.display();     //member function

        cout<<"\nemployees detail: \nid: "<<e1.getId()<<"\nname: "<<e1.getName() <<"\nsalary: 
"<<e1.getSalary()<<"\n";

        Employee e3;

        e3.display();

        e3.setId(401);   //member function

        e3.setName("sachin");  //member function

        e3.setSalary(60000);      //member function

        cout<<"\nafter setting values\n";

        e3.display();     //member function
```

```
            cout<<"\nemployees detail: \nid: "<<e3.getId()<<" \nname: "<<e3.getName()<<"\nsalary: 
"<<e3.getSalary()<<"\n";

            Employee e2(22,"pragati",50000);          //member function

            e2.display();      //member function

            printf("\ngetter\n");

            cout<<"\nemployees detail: \nid: "<<e2.getId()<<"\nname: "<<e2.getName()<<"\nsalary: 
"<<e2.getSalary()<<"\n";

            return 0;

}
```

3. Sales manager:

```
using namespace std;

#include<iostream>

#include<string.h>

struct SalesMan

{

            int id,target;

            char name[20];

            double salary,intensive;

            SalesMan()

            {

                        cout<<"\n\ndefault constructor called\n";

                        this->id=0;

                        strcpy(this->name,"not_given");

                        this->salary=0;

                        this->target=0;
```

```
{
        this->intensive=0;
}
SalesMan(int i,const char* n,double s,int t,int in)
{
        printf("\n\nparameterised constructor called\n");
        this->id=i;
        strcpy(this->name,n);
        this->salary=s;
        this->target=t;
        this->intensive=in;
}
void setId(int i)  //setters(mutator)
{
        this->id=i;
}
void setName(const char* n)                  //setters(mutator)
{
        strcpy(this->name,n);
}
void setSalary(double s)              //setters(mutator)
{
        this->salary=s;
}
void setTarget(int t)                  //setters(mutator)
{
        this->target=t;
}
void setIntense(double in)                  //setters(mutator)
{
        this->intensive=in;
}
int getId()                 //getters(accessor)
{
        return this->id;
}
char* getName()                 //getters(accessor)
{
        return this->name;
}
double getSalary()              //getters(accessor)
{
        return this->salary;
}
int getTarget()             //getters(accessor)
{
        return this->target;
}
double getIntense()             //getters(accessor)
{
        return this->intensive;
```

```
		}
		void display()
		{
			cout<<"\nsales managers details:\nid: "<<this->id<<"\tname: "<<this->name<<"\tsalary: 
"<<this->salary<<"\ttarget: "<<this->target<<"\tintensive: "<<this->intensive;
		}
};
int main()
{
		SalesMan m1;
		int id,target;
		char name[20];
		double salary,intensive;
		m1.display();	//member function
		cout<<"enter sale managers id:\n";
		cin>>id;
		cout<<"\nenter the name of sales manager:\n";
		cin>>name;
		cout<<"\nenter salary of sales manager:\n";
		cin>>salary;
		cout<<"\nenter target of sales manager:\n";
		cin>>target;
		cout<<"\nenter intensive for target completion:\n";
		cin>>intensive;
		m1.setId(id);	//member function
		m1.setName(name);			//member function
```

```
        m1.setSalary(salary);           //member function

        m1.setTarget(target);

        m1.setIntense(intensive);       //member function

        cout<<"\nafter setting values\n";

        m1.display();    //member function

        cout<<"\ngetter\nsales managers details:\nid: "<<m1.getId()<<"\tname: 
"<<m1.getName()<<"\nsalary: "<<m1.getSalary()<<"\ttarget: "<<m1.getTarget()<<"\tintensive: 
"<<m1.getIntense()<<"\n";

        SalesMan m3;

        m3.display();

        m3.setId(101);  //member function

        m3.setName("sachin");           //member function

        m3.setSalary(60000);            //member function

        m3.setTarget(40);

        m3.setIntense(4500);    //member function

        cout<<"\nafter setting values\n";

        m3.display();    //member function

        cout<<"\ngetter\nsales managers details:\nid: "<<m3.getId()<<"\tname: 
"<<m3.getName()<<"\nsalary: "<<m3.getSalary()<<"\ttarget: "<<m3.getTarget()<<"\tintensive: 
"<<m3.getIntense()<<"\n";

        SalesMan m2(22,"pragati",50000,45,4500);

        m2.display();    //member function

        cout<<"\ngetter\nsales managers details:\nid: "<<m2.getId()<<"\tname: 
"<<m2.getName()<<"\nsalary: "<<m2.getSalary()<<"\ttarget: "<<m2.getTarget()<<"\tintensive: 
"<<m2.getIntense()<<"\n";

        return 0;

}
```

4. Admin:

```
using namespace std;

#include<iostream>

#include<string.h>

struct Admin

{

	int id;

	char name[20];

	double salary,allowance;

	Admin()

	{

		cout<<"\n\ndefault constructor called\n";

		this->id=0;

		strcpy(this->name,"not_given");

		this->salary=0;

		this->allowance=0;

	}

	Admin(int i,const char* n,double s,double a)

	{

		cout<<"\n\nparameterised constructor called\n";

		this->id=i;

		strcpy(this->name,n);

		this->salary=s;

		this->allowance=a;

	}
```

```
void setId(int i)  //setters(mutator)

{

        this->id=i;

}

void setName(const char* n)                  //setters(mutator)

{

        strcpy(this->name,n);

}

void setSalary(double s) //setters(mutator)

{

        this->salary=s;

}

void setAllow(double a)             //setters(mutator)

{

        this->allowance=a;

}

int getId()                     //getters(accessor)

{

        return this->id;

}

char* getName()                      //getters(accessor)

{

        return this->name;

}

double getSalary()                   //getters(accessor)
```

```
	{
		return this->salary;
	}
	double getAllow()		//getters(accessor)
	{
		return this->allowance;
	}
	void display()
	{
		cout<<"\nadmins details:\nid: "<<this->id<<"\tname: "<<this->name<<"\tsalary: "<<this-
>salary<<"\tallowance: "<<this->allowance<<"\n";
	}
};
int main()
{
	Admin a1;
	int id;
	char name[20];
	double salary,allowance;
	a1.display();	//member function
	cout<<"enter admin id:\n";
	cin>>id;
	cout<<"\nenter name of the admin:\n";
	cin>>name;
	cout<<"\nenter salary of admin:\n";
	cin>>salary;
```

```
            cout<<"\nallowance for admin:\n";

            cin>>allowance;

            a1.setId(id);       //member function

            a1.setName(name);       //member function

            a1.setSalary(salary);       //member function

            a1.setAllow(allowance);//member function

            cout<<"\nafter setting values\n";

            a1.display();       //member function

            cout<<"\ngetter\n";

            cout<<"\nadmins details:\nid: "<<a1.getId()<<"\nname: "<<a1.getName()<<"\nsalary: 
"<<a1.getSalary()<<"\nallowance: "<<a1.getAllow()<<"\n";

            Admin a3;

            a3.display();

            a3.setId(101);    //member function

            a3.setName("sachin");   //member function

            a3.setSalary(60000);       //member function

            a3.setAllow(6000);         //member function

            cout<<"\nafter setting values\n";

            a3.display();       //member function

            cout<<"\ngetter\n";

            cout<<"\nadmins details:\nid: "<<a3.getId()<<"\nname: "<<a3.getName()<<"\nsalary: 
"<<a3.getSalary()<<"\nallowance: "<<a3.getAllow()<<"\n";

            Admin a2(101,"pragati",50000,4500);

            a2.display();       //member function

            cout<<"\ngetter\n";

            cout<<"\nadmins details:\nid: "<<a2.getId()<<"\nname: "<<a2.getName()<<"\nsalary: 
"<<a2.getSalary()<<"\nallowance: "<<a2.getAllow()<<"\n";
```

```
        return 0;
}
```

5. HR manager:

```
using namespace std;
#include<iostream>
#include<string.h>
struct HrManager
{
        int id;
        char name[20];
        double salary,commission;
        HrManager()
        {
                cout<<"\n\ndefault constructor called\n";
                this->id=0;
                strcpy(this->name,"not_given");
                this->salary=0;
                this->commission=0;
        }
        HrManager(int i,const char* n,double s,double c)
        {
                cout<<"\n\nparameterised constructor called\n";
                this->id=i;
                strcpy(this->name,n);
```

```
{
        this->salary=s;
        this->commission=c;
}
void setId(int i)  //setters(mutator)
{
        this->id=i;
}
void setName(const char* n)                 //setters(mutator)
{
        strcpy(this->name,n);
}
void setSalary(double s) //setters(mutator)
{
        this->salary=s;
}
void setComm(double c)                      //setters(mutator)
{
        this->commission=c;
}
int getId()                    //getters(accessor)
{
        return this->id;
}
char* getName()                        //getters(accessor)
{
```

```
                return this->name;
        }
        double getSalary()                    //getters(accessor)
        {
                return this->salary;
        }
        double getComm()        //getters(accessor)
        {
                return this->commission;
        }
        void display()
        {
                cout<<"\nHR Managers detail: \nid: "<<this->id<<"\tName: "<<this->name<<"\tSalary: 
"<<this->salary<<"\tCommission: "<<this->commission<<"\n";
        }
};
int main()
{
        HrManager h1;
        int id;
        char name[20];
        double salary,commission;
        h1.display();      //member function
        cout<<"\nenter hr managers id:\n";
        cin>>id;
        cout<<"\nenter name of hr manager:\n";
```

```
            cin>>name;

            cout<<"\nenter salary of hr manager:\n";

            cin>>salary;

            cout<<"\nenter commission for hr manager:\n";

            cin>>commission;

            h1.setId(id);       //member function

            h1.setName(name);       //member function

            h1.setSalary(salary);       //member function

            h1.setComm(commission);         //member function

            cout<<"\nafter setting values\n";

            h1.display();       //member function

            cout<<"\ngetter\nHR Managers detail: \nid: "<<h1.getId()<<"\nName:
"<<h1.getName()<<"\nSalary: "<<h1.getSalary()<<"\nCommission: "<<h1.getComm()<<"\n";

            HrManager h3;

            h3.display();

            h3.setId(101);    //member function

            h3.setName("sachin");   //member function

            h3.setSalary(60000);       //member function

            h3.setComm(6000);       //member function

            cout<<"\nafter setting values\n";

            h3.display();       //member function

            cout<<"\ngetter\nHR Managers detail: \nid: "<<h3.getId()<<"\nName:
"<<h3.getName()<<"\nSalary: "<<h3.getSalary()<<"\nCommission: "<<h3.getComm()<<"\n";

            HrManager h2(202,"pragati",50000,5000);          //member function

            h2.display();       //member function

            cout<<"\ngetter\nHR Managers detail: \nid: "<<h2.getId()<<"\nName:
"<<h2.getName()<<"\nSalary: "<<h2.getSalary()<<"\nCommission: "<<h2.getComm()<<"\n";
```

```
        return 0;
}
```

6. Date:

```
using namespace std;
#include<iostream>
struct Date
{
        int day,month,year;
        Date()
        {
                cout<<"\n\ndefault constructor called\n";
                this->day=0;
                this->month=0;
                this->year=0;
        }
        Date(int d,int m,int y)
        {
                cout<<"\n\nparameterised constructor called\n";
                this->day=d;
                this->month=m;
                this->year=y;
        }
        void setDay(int d)          //setter(mutator)
        {
```

```
        this->day=d;
}
void setMonth(int m)    //setter(mutator)
{
        this->month=m;
}
void setYear(int y)                 //setter(mutator)
{
        this->year=y;
}
int getDay()      //getters(accessor)
{
        return this->day;
}
int getMonth()             //getters(accessor)
{
        return this->month;
}
int getYear()              //getters(accessor)
{
        return this->year;
}
void display()
{
        cout<<"\n\ndate is: \n"<<this->day<<"/"<<this->month<<"/"<<this->year<<"\n";
```

```
            }
};
int main()
{
            Date d1;
            int day,month,year;
            d1.display();       //member function
            cout<<"\nenter date: ";
            cin>>day;
            cout<<"\nenter month: ";
            cin>>month;
            cout<<"\nenter year: ";
            cin>>year;
            d1.setDay(day);             //member function
            d1.setMonth(month);                 //member function
            d1.setYear(year);                   //member function
            cout<<"\nafter setting values\n";
            d1.display();       //member function
            cout<<"\ngetter\ndate: "<<d1.getDay()<<"\nmonth: "<<d1.getMonth()<<"\nyear: 
"<<d1.getYear()<<"\n";
            Date d3;
            d3.display();
            d3.setDay(4);               //member function
            d3.setMonth(10);                    //member function
            d3.setYear(2018);                   //member function
            cout<<"\nafter setting values\n";
```

```
	d1.display();	//member function
	cout<<"\ngetter\ndate: "<<d1.getDay()<<"\nmonth: "<<d1.getMonth()<<"\nyear: 
"<<d1.getYear()<<"\n";
	Date d2(23,4,2001);			//member function
	d2.display();
	cout<<"\ngetter\ndate: "<<d2.getDay()<<"\nmonth: "<<d2.getMonth()<<"\nyear: 
"<<d2.getYear()<<"\n";
	return 0;
}
```

7. Time:

```
using namespace std;
#include<iostream>
struct Time
{
	int hr,min,sec;
	Time()
	{
		cout<<"\n\ndefault constructor called\n";
		this->hr=-1;
		this->min=-1;
		this->sec=-1;
	}
	Time(int h,int m,int s)
	{
		cout<<"\n\nparameterised constructor called\n";
```

```
        this->hr=h;

        this->min=m;

        this->sec=s;

}

void setHour(int h)                //setter(mutator)

{

        this->hr=h;

}

void setMin(int m)                 //setter(mutator)

{

        this->min=m;

}

void setSec(int s)                 //setter(mutator)

{

        this->sec=s;

}

int getHr()                //getter(accessor)

{

        return this->hr;

}

int getMin()       //getter(accessor)

{

        return this->min;

}

int getSec()       //getter(accessor)
```

```
        {
                return this->sec;
        }
        void display()
        {
                cout<<"\ntime is: "<<this->hr<<":"<<this->min<<":"<<this->sec;
        }
};
int main()
{
        Time t1;
        int hr,min,sec;
        int r,q;
        t1.display();       //member function
        cout<<"\nenter hours:\n";
        cin>>hr;
        cout<<"\nenter minuits:\n";
        cin>>min;
        cout<<"\nenter seconds:\n";
        cin>>sec;
        if(sec>=60)
        {
                r=sec%60;
                q=sec/60;
                sec=r;
        min=min+q;
}
if(min>=60)
{
        r=min%60;
        q=min/60;
        min=r;
        hr=hr+q;
}
t1.setSec(sec);             //member function
t1.setMin(min);             //member function
t1.setHour(hr);             //member function
cout<<"\nafter setting value\n";
t1.display();       //member function
cout<<"\ngetter\nhour: "<<t1.getHr()<<"\nmin: "<<t1.getMin()<<"\nsec: "<<t1.getSec()<<"\n";
Time t3;
t3.display();
t3.setSec(40);              //member function
t3.setMin(56);              //member function
t3.setHour(7);              //member function
cout<<"\nafter setting value\n";
t3.display();       //member function
cout<<"\ngetter\nhour: "<<t3.getHr()<<"\nmin: "<<t3.getMin()<<"\nsec: "<<t3.getSec()<<"\n";
Time t2(10,49,55);                  //member function
t2.display();       //member function
```

```
	cout<<"\ngetter\nhour: "<<t1.getHr()<<"\nmin: "<<t1.getMin()<<"\nsec: "<<t1.getSec()<<"\n";

	return 0;

}
```

8. Distance:

```
using namespace std;

#include<iostream>

struct Distance

{

	int feet,inch;

	Distance()

	{

		cout<<"\n\ndefault constructor called\n";

		this->feet=-1;

		this->inch=-1;

	}

	Distance(int f,int i)

	{

		cout<<"\n\nparameterised constructor called\n";

		this->feet=f;

		this->inch=i;

	}

	void setFeet(int f)		//setter(mutator)

	{

		this->feet=f;
	}
	void setInch(int i)			//setter(mutator)
	{
		this->inch=i;
	}
	int getFeet()	//getter(accessor)
	{
		return this->feet;
	}
	int getInch()	//getter(accessor)
	{
		return this->inch;
	}
	void display()
	{
		cout<<"\ndistance is: "<<this->feet<<"feet and "<<this->inch<<"inches\n";
	}
};
int main()
{
	Distance d1;
	int feet,inch;
	d1.display();	//member function
	cout<<"\nenter distance in feet:\n";
	cin>>feet;
```

```
        cout<<"\nenter distance in inch:\n";
        cin>>inch;
        d1.setFeet(feet);        //member function
        d1.setInch(inch);        //member function
        cout<<"\nafter setting values\n";
        d1.display();    //member function
        cout<<"\ngetter\nfeet: "<<d1.getFeet()<<"\ninch: "<<d1.getInch()<<"\n";
        Distance d3;
        d3.display();
        d3.setFeet(6);   //member function
        d3.setInch(2);   //member function
        cout<<"\nafter setting values\n";
        d3.display();    //member function
        cout<<"\ngetter\nfeet: "<<d3.getFeet()<<"\ninch: "<<d3.getInch()<<"\n";
        Distance d2(5,2);        //member function
        d2.display();    //member function
        cout<<"\ngetter\nfeet: "<<d2.getFeet()<<"\ninch: "<<d2.getInch()<<"\n";
        return 0;
}
```

9. Complex:

```
using namespace std;
#include<iostream>
struct Complex
{
}

int real,imag;

Complex()

{

        printf("\n\ndefault constructor called\n");

        this->real=0;

        this->imag=0;

}

Complex(int r,int i)

{

        printf("\n\nparameterised constructor called\n");

        this->real=r;

        this->imag=i;

}

void setReal(int r)                 //setters(mutator)

{

        this->real=r;

}

void setImg(int i)          //setters(mutator)

{

        this->imag=i;

}

int getReal()               //getters(accessor)

{

        return this->real;

}
```

```
        int getImag()    //getters(accessor)
        {
                return this->imag;
        }
        void display()
        {
                cout<<"\ncomplex number: %d+%di\n",this->real,this->imag;
        }
};
int main()
{
        Complex c1;
        c1.display();    //member function
        int real,imag;
        cout<<"\nenter real part of complex number:\n";
        cin>>real;
        cout<<"\nenter imaginary part of complex number:\n";
        cin>>imag;
        c1.setReal(real);        //member function
        c1.setImg(imag);         //member function
        cout<<"\nafter setting values\n";
        c1.display();            //member function
        cout<<"\ngetter\n";
        cout<<"\ncomplex number: "<<c1.getReal()<<"+"<<c1.getImag()<<"i"<<"\n";
        Complex c3;
        c3.display();
        c3.setReal(15);  //member function
        c3.setImg(3);     //member function
        cout<<"\nafter setting values\n";
        c3.display();                    //member function
        cout<<"\ngetter\n";
        cout<<"\ncomplex number: "<<c3.getReal()<<"+"<<c3.getImag()<<"i"<"\n";
        Complex c2(10,2);          //member function
        c2.display();      //member function
        cout<<"\ngetter\n";
        cout<<"\ncomplex number: "<<c2.getReal()<<"+"<<c2.getImag()<<"i\n";
        return 0;
}
```