//// answer-1
//object passing code

```
#include<iostream>
using namespace std;

class student{
        int rollnum,marks1,marks2;
        public:
                void set(int a,int b,int c){
                        rollnum=a;
                        marks1=b;
                        marks2=c;
                }
                void get()
                {
                    cout<<rollnum<<endl<<marks1<<endl<<marks2<<endl;
                }

                void addition(student s1,student s2){ // parameterized constructor
                        marks1=s1.marks1 + s2.marks1;
                        marks2=s1.marks2 + s2.marks2;
                        rollnum=3;

                }
};

int main(){
        student obj1,obj2,obj3;
        obj1.set(1,99,2);
        obj2.set(2,1,98);
        obj3.set(0,0,0);

        obj3.addition(obj1,obj2);       // object passing
    obj3.get();
        return 0;
}
```

//answer-2
//Friend fun code

```
#include<iostream>
using namespace std;

class subject2;                         // forward declaration
class total;                    // forward declaration

class subject1{
        int rollnum,theory,practical;
        public:
        void set(int a,int b,int c){
                rollnum=a;
                theory=b;
                practical=c;
        }

        friend total findsum(subject1,subject2);
};

class subject2{
        int rollnum,theory,practical;
        public:
        void set(int a,int b,int c){
                rollnum=a;
                theory=b;
                practical=c;
        }
        friend total findsum(subject1,subject2);
};

class total{
        int rollnum,theoryTotal,practicalTotal;
        public:
        void set(int a,int b,int c){
                rollnum=a;
                theoryTotal=b;
                practicalTotal=c;
        }

        friend total findsum(subject1,subject2);

        void display(){
                cout<<"Roll Number : "<<rollnum<<endl
                <<"theoryTotal : "<<theoryTotal<<endl
                <<"practicalTotal : "<<practicalTotal<<endl;
        }
};

total findsum(subject1 s1,subject2 s2){
                total temp;
                temp.rollnum=1;
                temp.theoryTotal=s1.theory + s2.theory;
                temp.practicalTotal=s1.practical + s2.practical;

                return temp;
}

int main(){
        subject1 s1;
        subject2 s2;
        total t1;
        s1.set(1,5,10);
        s2.set(1,95,90);
        t1.set(1,0,0);

        t1=findsum(s1,s2);
        t1.display();

        return 0;
}
```

Q3) What is a friend function. When is it used? How is it different from a member function? *
answer) A friend fun is present in one or more classes but it not a member of any class. It is only present as a friend function.
It is used when you want to access private variables of several classes in a function.
Member fun is a member of only one class.
Member fun is always called using an object.

Q4) What happens when you declare a function Inline? When do you declare a function Inline? Why do you use an Inline function? *
A) It will be expanded rather than called.
When you want to save fun call overhead.
To save overhead associated with a fun call.

Q4) Inline code:

```
/* Functions defined inside a class always inline
so there is no need to write inline while defining the functions inside the class */

#include <iostream>
using namespace std;
class operation
{
        int a,b;

public:
        void set(){
        cout<<"Enter two values : ";
        cin>>a>>b;
        }

        void sum(){

        cout << "Addition of two numbers : " << a+b << "\n";
        }
};
```

```
int main(){
	cout << "Program using inline function\n";
	operation s;
	s.set();
	s.sum();

    return 0;
}
```

Q5) What happens when you declare a variable as static? *
a) All objects will share same copy of that variable.

Q6) Create a student class with constructor and destructor. Include a count variable to count number of objects created and destroyed using constructor and destructor. Create 5 objects. When each object is created count should be displayed. When an object is destroyed again new updated count should be displayed. *

A)

```
//answer-6
#include<iostream>
using namespace std;

class test{
	static int count;
	public:
		test(){
			count++;
			cout<<"Object created. Count= "<<count<<endl;
		}

		~test(){
			count--;
			cout<<"Object destroyed. Count= "<<count<<endl;

		}
};
int test :: count;
int main(){
        test c1,c2;

        while(1)// local block 1
    {
       test c3, c4;
       break;
    }
        test c5, c6;
        if(1) // local block 2
    {
       test c7, c8;
    }
        test c9, c10;

        return 0;
}
```

Q7) Why do you use this pointer? *
a) This pointer has several uses. For eg. If there is an ambiguity due to same name of object variables and fun parameters, then obj variables are hidden. In such case they are accessed by this ptr.

Q8) code:

```
//8 this pointer
#include <iostream>
using namespace std;

class Employee {
          int id;
      string name;
      float salary;

          public:
        Employee(int id, string name, float salary){
                        this->id = id;
               this->name = name;
                        this->salary = salary;
```

```
                }

            void display(){
                cout<<id<<"  "<<name<<"  "<<salary<<endl;
            }
};

int main(void) {
   Employee e1(1,"Anuj",10000);
   Employee e2(2,"Tanuj",20000);
   e1.display();
   e2.display();
   return 0;
}
```

Q9)WAP to show use of pointer to objects. Create a student class. Create two objects. Assign, print, add values of these two objects using pointer to object. *
A)

```
#include<iostream>
using namespace std;
class student
{
   private:
   int rn,m1,m2;
   public:
   void set(int, int, int);
   void get();
   void add();
};

void student::set(int x, int y, int z)
{
   rn=x;
   m1=y;
   m2=z;
}

void student::add()
```

```
{
   cout<<m1+m2<<endl;
}

void student::get()
{
   cout<<rn<<endl<<m1<<endl<<m2<<endl;
}

int main()
{
   student obj1;
   student *ptr1;
   ptr1=&obj1;
   ptr1->set(1,50,70);
   ptr1->get();
   ptr1->add();

   student obj2;
   student *ptr2;
   ptr2=&obj2;
   ptr2->set(1,2,8);
   ptr2->get();
   ptr2->add();



}
```

```
Q10)//qs-3
#include<iostream>
using namespace std;
class library
{
        string author;
        string title;
        public:
                void display()
                {

                                cout<<"AUthor = "<<author<<endl;
                                cout<<"Title = "<<title<<endl;

                }
                library()
                {
                }
                library(string x,string y)
                {

                   author=x;
              title=y;

                }
                char findbyauthor(string z )
                {
                    if(author==z)
           {
                                return 'y';

                        }
                        else return 'n';

                }
                char findbytitle(string u)
                {
                        if(title==u)
                     {
                        return 'y';

                        }

                        else return 'n';

                 }
};
int main()
{
        library obj[5]  = { library("author1","title1"),
                    library("author2","title2"),
                    library("author3","title3"),
                    library("author4","title4"),
                    library("author4","title5")

        };

        string author, title;
        cout<<"displaying all book and authors"<<endl;
        int i;
        for(i=0;i<5;i++)
        {

          obj[i].display();

        }

        cout<<"Enter the author you want to find "<<endl;
        cin>>author;
        char found='n';
        for(i=0;i<5;i++)
   {
      found=obj[i].findbyauthor(author);
      if(found=='y') break;

   }
```

```
    if(found=='y')
        cout<<"The input author "<<author<<" is **available** in lib"<<endl;
    else
        cout<<"The input author "<<author<<" is not available in lib"<<endl;



    cout<<"Enter the title you want to find "<<endl;
    cin>>title;
    found='n';
         for(i=0;i<5;i++)
    {
        found=obj[i].findbytitle(title);
        if(found=='y') break;
    }
    if(found=='y')
        cout<<"The input title "<<title<<" is **available** in lib"<<endl;
    else
        cout<<"The input title "<<title<<" is not available in lib"<<endl;

}
```