Introduction to OOPs

Q1 How did the evolution of programming languages occured?

A a) Machine Code:

- M/c code is at the lowest level. It is close to the hardware.
- Initially programmers used to write in m/c code which is strictly binary.
  For eg:- 100 01 11 to add.
- It becomes difficult to understand & write.

b) Assembly Language

- Simple words were added like
  add to 01
- Memory management still done by the programmer.

c) Structured programming

- Advanced tools like compiler, linker for m/m mgmt.
- Easy to u/stand & use.
- More features like - function, loops etc.

d) OOP languages
- Based on real world entities.
- More organised
- Use of classes. Main focus on data/var.

Q What is the core concept of OOPS?
A Classes.

Q What is a class?
A A class is a collection of var & fun.
It is a blueprint for creating objects.

Q Why are var & fun combined into classes?
A
- Bcoz it is based on real world entities
For eg:- Fridge has some items like food, drink, juice and it has some fun like open_door, turn_on, turn_off.
- So these variables & fun r combined into classes.
- Fun shud be related to the class

Q What is the significance of classes?
A It provides all the features of OOPs like
1) encapsulation
2) Inheritence
3) Polymorphism
4) Data hiding
5) Abstraction
6) Classes
7) Objects

Q What are objects?

A Object is an instance of the class. Objects r created to make use of a class. Each object will have same var & fun, but values of variables may be different.

Q How do you use classes?

A Classes are used with the help of objects.

Q C vs C++.

C	C++
• Focus is on fun.	• Focus is on var/data.
• Procedure oriented lang.	• object oriented
• There r no classes. Only fun.	• Both classes & fun.
• No namespaces	• Namespaces.

Class Specification

- Class specification means how many variables & function shud be there in class. It means following.

1) How many variables shud be there & their data types.
2) How many fun. shud be there
3) How many parameters shud fun have.
4) what shud be return type of fun.

Eg 1) Create a class with 2 var - a, b.
3 fun - set() to assign values
- get() to print "
- add() " add "

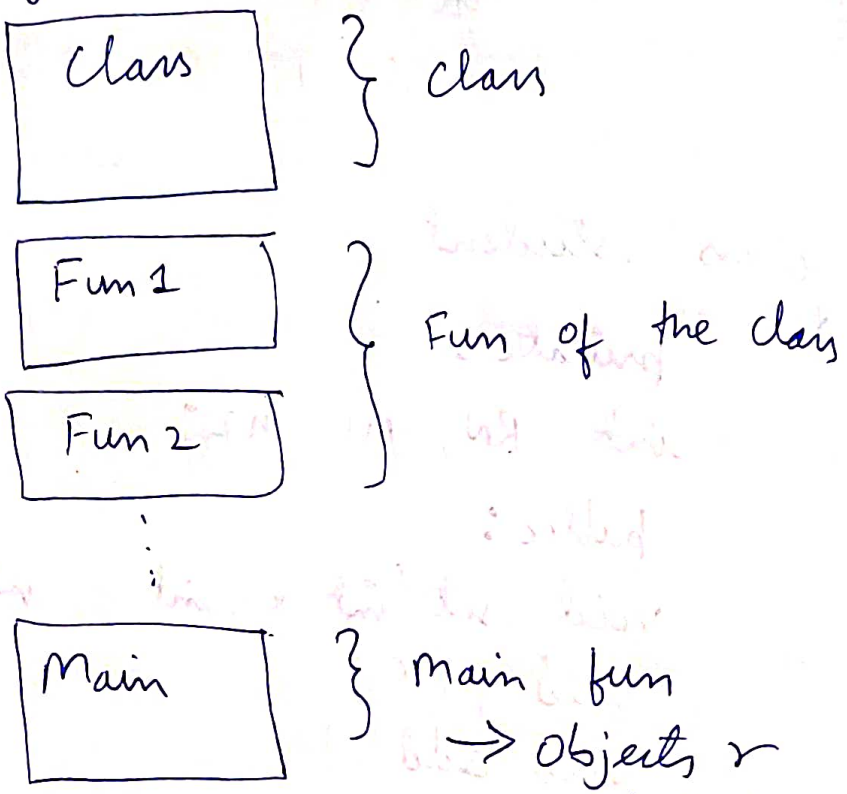
```
class myclass
{
    private:
    int a, b;
    public:
    void set(int x, int y);// 2 Parami
    void get(); // No param.
    void add(); //

};
```

Eg 2) Create a class student with
3 var :- rollno, marks1, marks2
3 fun :- set ( )
get ( )
add ( )

A)
```
class student
{
    private:
    int RN, M1, M2;
    public:
    void set(int x, int y, int z)// This will have 3 Param not 2.
    void get ( )
    void add ( )
};
```

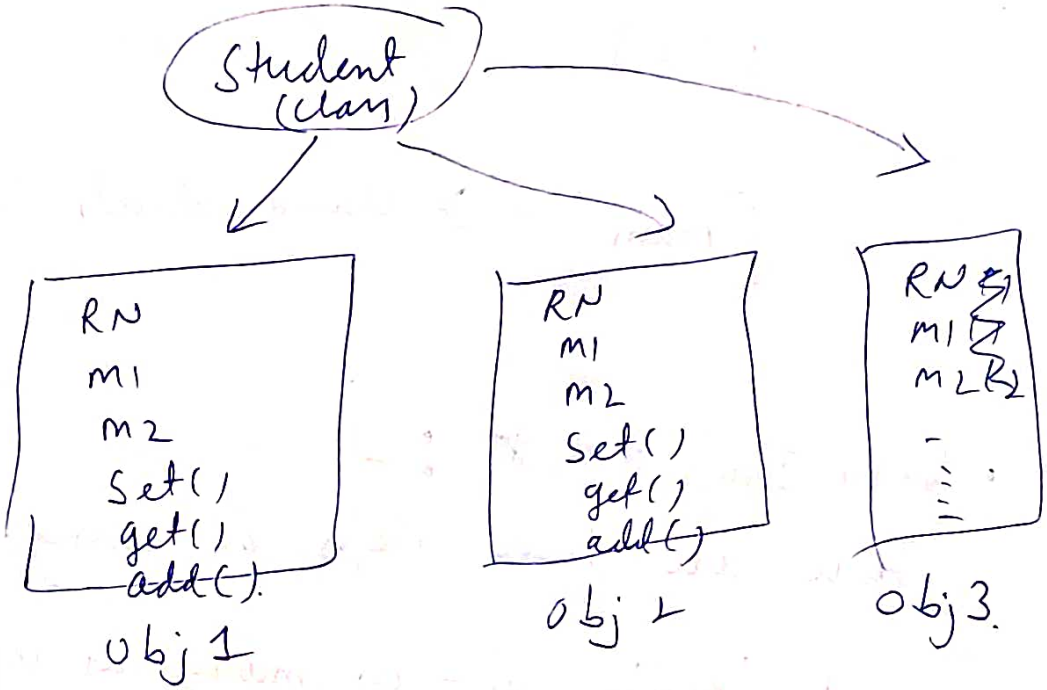
The complete code

- Any code with classes will have three regions :



Using objects

1) When you create a class, it is used by the help of objects
2) You need to create obj to use class.
3) Each obj will have same var & fun wh. r present in the class.



4) Objects r always created inside the main fun.

How to use objects

- Before using objects, it is assumed that class is already created.
- Objects r always used inside main.

- V.V. Imp ★★★ :-
  There are four steps in using objects.

Step 1) Create obj - as many as required
- doesn't depend on variables.

Step 2) Assign values - to variables of obj.
- use set() fun.

Step 3) Print values - of var of obj
- use get() fun.

Step 4) add values.

Q) Write a code to use objects.
(It is not the complete code, only main fun is written).

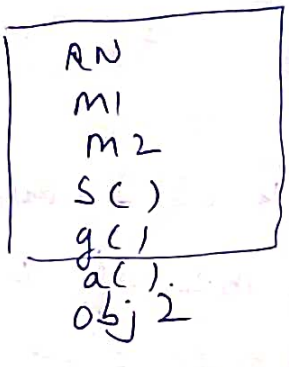
A)
```
int main()
{
```

Step ① student obj1, obj2 // create objects.

↓ class name ~ obj name.

Step ②

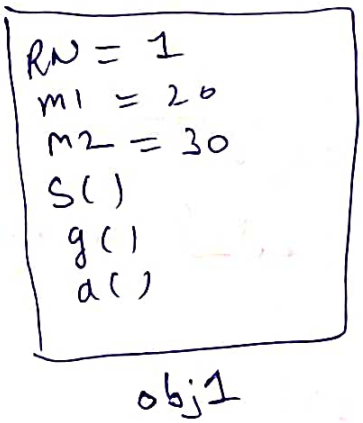
m1 → marks 1
m2 → marks 2.

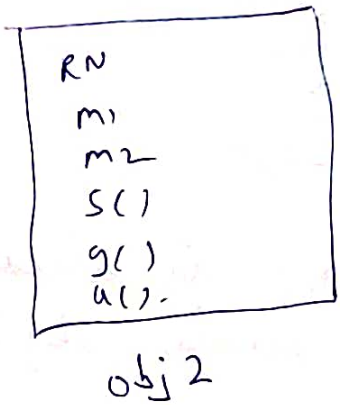


S() is set fun
g() is get fun
a() is add fun.

- Two obj r created. They are blank. ~~Here~~ RN, m1, m2 have no values. There will be no output.

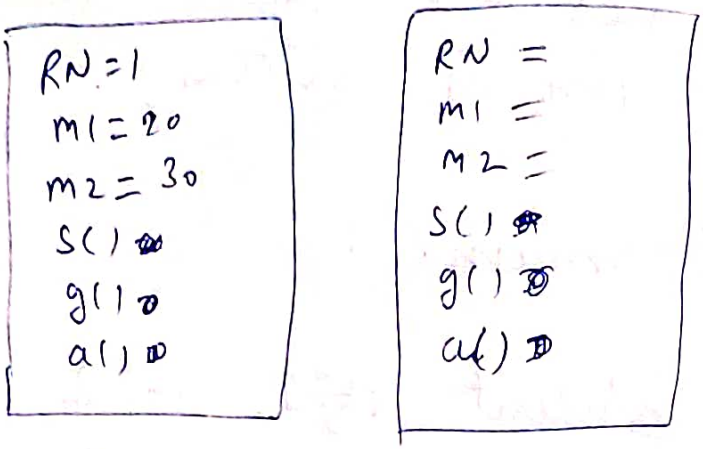
Step-2 ② obj1.set(1, 20, 30) // assign value to obj 1.





- Set fun is used to assign value to obj1. obj2 is empty. Values r assigned to obj1 and not to class. Nothing will be done in class, everything is done in obj.

Step-3 ③ obj1.get() //



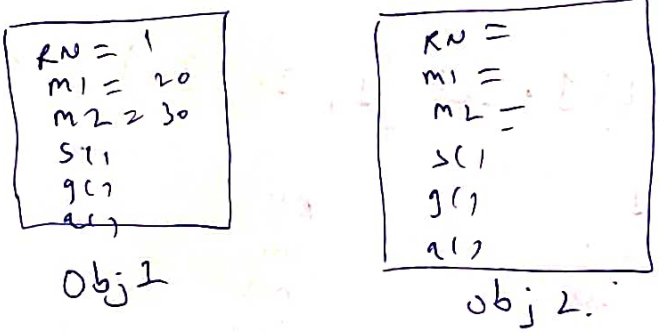
obj 1 obj 2.

- Values of obj 1 will be printed.
  For eg 1, 20, 30.

Step-4 ④ obj1.add()

- M1 & m2 of obj1 will be added.
- ~~30~~ (20+30=)50 will be printed.

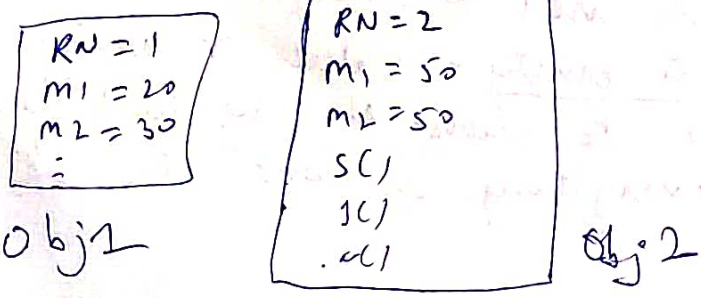
Now do these steps for obj2:



obj1 obj2.

① // obj2 is already created. No need to create obj2.

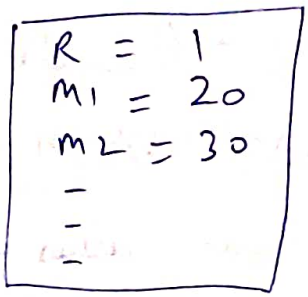
② obj2.set(2, 50, 50)



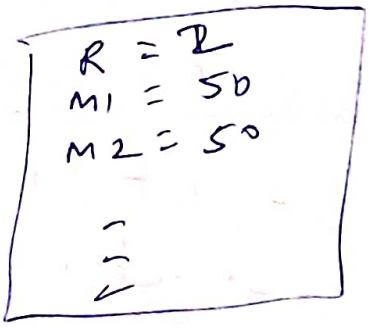
obj1 obj2

. Now values r assigned to obj 2.

(3) Obj 2. get ( )



Obj 1



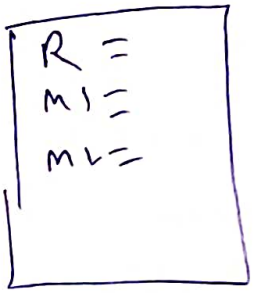
Obj 2

~~2 50 50 will~~

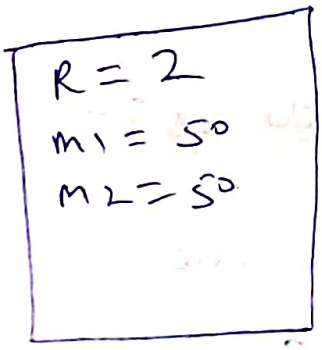
. 2
50
50

will be printed . There r values of obj 2.

(4) obj 2. add ( )



obj 1



Obj 2

100 will be printed becoz sum of 50 & 50 in obj 2 is 100.

(5) ∴ one.

Summary: How to use objects

```
int main( )
{
1) student obj1, obj2; // Create obj.
2) obj1.set (1, 20, 30); // assign value to obj1
3) obj1.get ( ); // Print values of obj1
4) obj1.add ( ); // add " " "

5) obj2.set (2, 50, 50); // assign obj2
6) obj2.get ( ); // Print obj2
7) obj2.add ( ); // added obj2.

}
```

Output will be :-

```
1
20   } line 3
30
50   } line 4

2
50   } line 6
50
100. } line 7
```

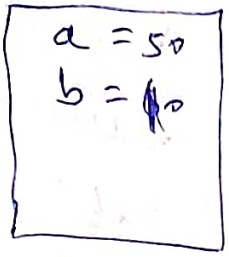
Eg-2 Create a class named calculator with 2 var - a, b & following fun

1) set() to assign values.
2) get() to print
3) add() to add values
4) Mul() to multiply
5) Div() to divide
6) sub() to subtract.

- Create two objects with values (50, 10) (100, 20) & perform, add sub, mul..

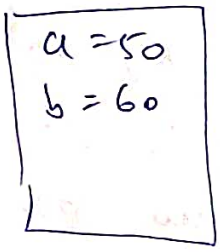
ans
```
class calculator
{
    private:
    int a, b;
    public:
    void set (int x, int y);
    //There r 2 var, so set fun will
      have only two parameters.
    void get ();
    void add ();
    void mul ();
    void div ();
    void sub ();
};
```

```
int main ( )
{
1) * Calculation  obj1, obj2 ;
     ↓              ↓    ↓
   Class name     obj names
2) obj1.set (50,10);
3) obj1.get ( );
4) obj1.add ( );
5) obj1.sub ( );
6) obj1.mul ( );
7) obj1.div ( );

8) obj2.set (100, 20);
9) obj2.get ( );
10) obj2.add ( );
11) obj2.sub ( );
12) obj2.mul ( );
13) obj2.div ( );

}
```

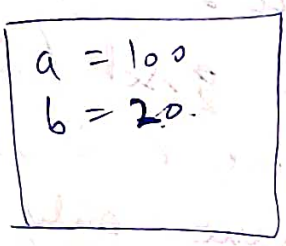


obj1

obj2



obj1



obj2

Explanation :-

Line :- 1 - obj1 & obj2 r created. Both r blank.
2 - obj1 is given values 50, 10.
3 - obj1 is printed - i.e 50, 10.
4 - 50, & 10 r added
5 - 50 & 10 r subtracted
6 - 50 & 10 r multiplied
7 - 50 & 10 r divided.

8 - obj2 given values - 100, 20
9 - obj2 is printed
10 - ~~obj2~~ ~~is~~ 100 & 20 r added
11 - " --- sub.
12 - ------- mul
13 - ------- div.

Output :-

50 } line-3
10 }
60 } line-4
40 } ----5
500 } ---6
5 } ----7

100 } ---9
20
120 ⟶ 10
80 ⟶ 11
2000 ⟶ 12
5 ⟶ 13.