OOPS 1

Hello Everyone

Agenda:

1) Programming Paradigms
2) Procedural programming
3) Object oriented programming
4) Access Modifiers.

Programming Paradigm: Style/standard way of writing program.

Without programming paradigm code will be:

1) hard to read and understand
2) hard to test and maintain
3) less structured.

Type of programming paradigms?

1) Imperative programming
2) Procedural Programming
3) object oriented Programming
4) Functional Programming.

Imperative Programming: Telling computer how to do task by giving a set of instructions in a particular order i.e line by line.

Imperative Programming eg.

```
// for eg.
int a = 10;
int b = 20;
int sum = a+b;
print(sum);
int diff = a-b;
print(diff);
```

Procedural Programming: Split the program into multiple function (section of code that performs a specific task) which are reusable code blocks.

* Procedural Programming :-

```
// Procedural programming eg
    int a = 10;
    int b = 20;
    addTwoNumber(a,b);
    subtractTwoNumber (a,b);

 void addTwoNumber (a,b)
{
      int sum = a+b;
       Print (sum);
}

 void subtractTwoNumber (a,b)
{
      int diff = a-b;
       Print (diff);
}
```

*) Declarative programming: Specify what you want program to do, without specifying how should it be done.

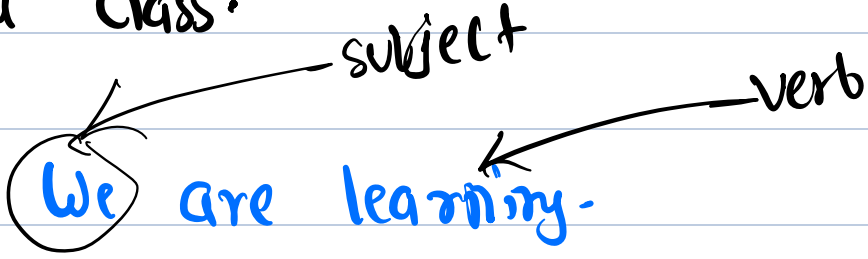
```
select * from tableName
```

Procedural Programming:

It splits entire program into small procedure or function which are reusable.

Eg. C.

⇒ In one line, what we are doing in the class.



1) We are learning.
2) We are learning structure programming.
3) We are hiding our privacy.
4) Vishal is teaching.

Subject + verb.
Entity performs action.

```
PrintStudent ( String name, int age, String gender)
{
    Print (name);
    Print (age);
    Print (gender);
}
```

Combine set of Attribute : Struct or Structure.

```
Struct Student
{
    String name;
    int age;
    String gender;
}
```

```
PrintStudent ( Student st)
{
    Print (st.name);
    Print (st.age);
    Print (st.gender);
}
```

→ Who is someone : Student
Who is something: PrintStudent

* All variable declared in structure are public
↳ Data Privacy issue.

Cons of procedural programming:

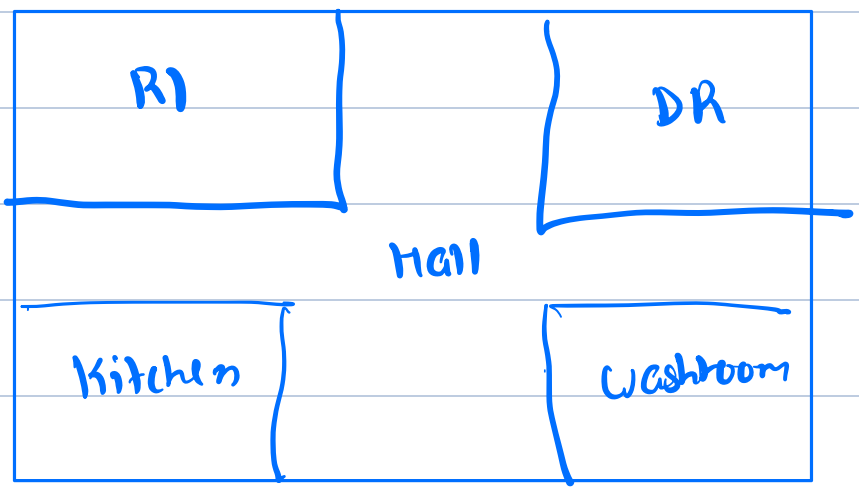
1) Privacy
2) Reliability
3) difficult to debug and understand

OOPS:

1) Entity are core in OOPS.
2) entity can have attribute and behavior.

Class: Blueprint of an idea.

Eg. Flow plan for home.




```
Class student
{   String name;
    int age;
    String gender;
    double PSP;
    attendClass();
    giveContest();
    solveProblem();
}
```

1) Class takes no space in memory.
2) Multiple instance of same class can be created.

* Object of class will occupy memory because they are actual instance.

```
public class Student
{
    String name;
    String batchName;
    int age;
    double psp;

    void changeBatch(String newBatch)
    {
        batchName = newBatch;
    }

    void giveContest()
    {
        Print(" Giving Contest");
    }
}
```

attribute, data member

Student s1 = new Student();

Pillar of OOPS:

1) Principle - 1 — abstraction
2) Pillars - 3

Fundamental principle / concept

Support to hold the things together

1) Inheritence
2) Polymorphism
3) Encapsulation.

Principle: I want to be a good person.

pillar: 1) I will be truthful
2) I will do hard work
3) respect everyone.

Abstraction: Representing some idea.

What is the main purpose of abstraction:

→ Don't need to know the details of idea.

1) Data
2) Behavior

Encapsulation: Capsule

Try to open capsule

1) It will flow away. So first purpose is to hold the medicine powder together.
2) It help to avoid mixing with each other
3) protect from outside environment.

What we really store together in prog.

⇒ attribute and behaviour.

Advantage:

1) It holds attribute and behaviour together.
2) protect member from illegitamet access.

→ Access modifier

Type of Access modifiers:

1) **Public** : Can access everywhere.

2) **Private** : Can be accessed only inside class definition.

3) **Protected** : Can be accessed only from classes of some pkg or subclass can access.

4) **Default** : Can be accessed in same class and same package.

	Class	Package	Subclass (same package)	Subclass (diff PKG)	World
Public	Yes	Yes	Yes	Yes	Yes
protected	Yes	Yes	Yes	Yes	No
no modifier	Yes	Yes	Yes	No	No
Private	Yes	No	No	No	No

Quiz:

Which is more restricted access modifier :

Private

Quiz:

Which is the most open access modifier?

Public

This Keyword:

1) This keyword is used to refer current instance of class or object.

```
Public class person
{
    Private String name;

    Public Person (String name)
    {
        this.name = name;
    }

    public void introduceYourself (String name)
    {
        print (" Hello, I am " + this.name);
    }

main()
{
    Person person1 = new Person ("Alice");
    Person person2 = new Person ("Bob");

    person1.introduceYourself();

    person2.introduceYourself();


    }

}
```

1) distinguish bet^n instace variable (object) and local variable. (method parameter inside fun)

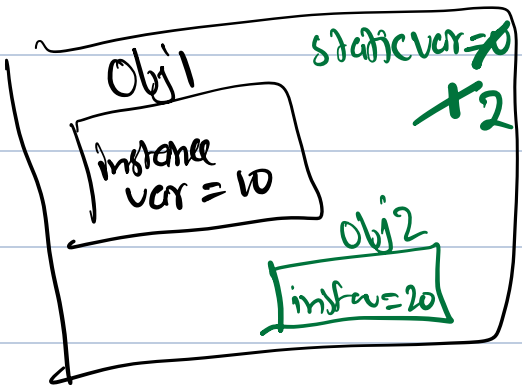
2) To avoid variable name ambiguity.

Static Keyword: Static Keyword in Programming language are used to declare class-level member or method, which are associated with class itself, instead of object or instance of class.

* **Static variable:** These variables are shared among all objects. They are initialized once when the class loads, their value is common to all object.

* **Static method:** Method invokes by the class. They can access static variable and perform operation that doesn't required access to instance specific (object) data.

```
public class MyClass
{
    // static variable
    static int staticVar = 0;

    // instance variable
    int instanceVar;

    public MyClass(int value)
    {
        this.instanceVar = value;
        staticVar++;
    }

    // main method
    main()
    {
        MyClass obj1 = new MyClass(10);
        MyClass obj2 = new MyClass(20);

        Print("static variable : " + staticVar);
        //output : 2

        Print("Instance variable1: " + obj1.
                                    instanceVar);
        //output: 10

        Print("Instance variable2: " + obj2.
                                    instanceVar);
        //output: 20
```



Scope of Variable

1) Class/static scope
2) Instance scope
3) function scope
4) Block scope.

1) Class level scope: variable declared with static keyword will be having class level scope. The variable are associated with class.

2) Instance scope: variable declared within the class but outside any method or constructor would be instance scope.

3) **Method Scope:** Variable declared inside method.

4) **Block Scope:** variable declared within pair of curly braces.

```
public class ScopeExample
{
    // Class level scope
    static int classVar = 10;

    // instance level scope
    int instanceVar = 20;


    public void method1()
    {
        // method level scope
        int methodVar = 30;


        if(true)
        {
            // block level scop
            int blockVar = 40;
            print (classVar + instanceVar
                + methodVar + blockVar);
        //
        }
        //
    }

    main()
    {
ScopeExample Obj = new ScopeExample();
    Obj. exampleMethod();

    //


    }
```

Doubt session

Blog
- viewer
- blogger Info,
- admin

Banking College Employment

IRCTC.