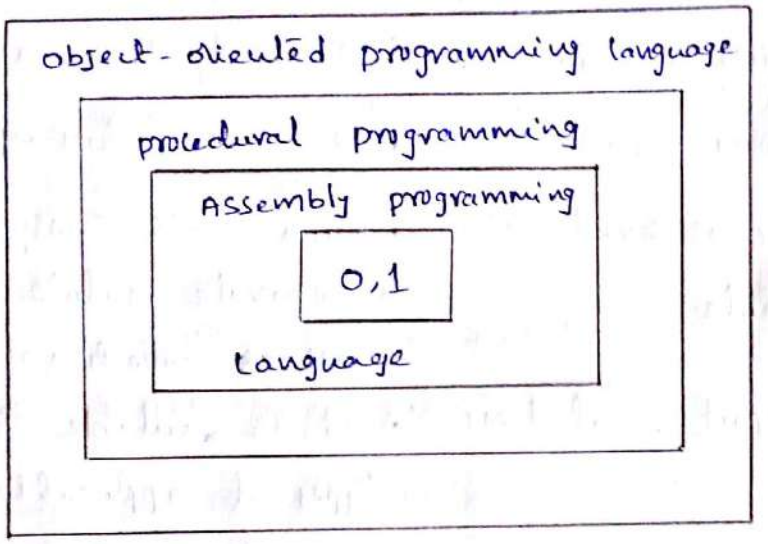
* Introduction *

* Language: It is a communication barrier, used by us to communicate with each other. In terms of programming Language is a barrier between human & a system or application (software).



→ Languages broadly classified as

- Low level programming language
- Middle level programming language
- High-level programming language

→ In low level programming language, to give instructions to a machine, we use Assembly language. It comprises of Mnemonics.

→ Next, we used middle level programming language for communication. The best known example for this is 'C'. It comes under procedure oriented programming language. It uses well structured steps and procedures to build a program. In simple terms, it is a collection of functions or procedures. Mostly, it uses english words as their identifiers.

→ Next, We started using high level programming languages for communication, the best example for this JAVA. It comes under object oriented programming language.

* Differences between procedure oriented / structure oriented and object oriented programming

procedure oriented programming language	object oriented programming language
• It mainly focus on "process".	• It mainly focus on "data".
• It uses top-down Approach	• It uses bottom-up approach.
• Each function considered as a separate module.	• Each class considered as a separate module, where class is collection of Methods.
• It doesn't support real-time applications.	• It is suitable for all types of applications.
• It is difficult to debug an application and extend any application.	• It is easy to debug and extend any application.
• It doesn't provide any security Ex:- 'C'	• It provides security Ex:- 'Java'
• Less reusability	• More reusability

→ Top-down approach is also called as step-wise approach. In point of 'C', this approach first programmer has to write a code for main function, In that, they will call sub-functions.

→ Bottom-up approach starts with low-level system, then it looks for high level system. In this, first programmer has to write code for modules, then they look for integration of modules.

* Object oriented Thinking :-

→ Everywhere in the real world we can see objects like people, animals, plants, cars, buildings and computers and soon.

→ Sometimes, we divide objects into two types, those are Animate and Inanimate.

→ Animate objects are "alive" in some sense they move around and do things. Inanimate objects, on the other hand, do not move on their own.

→ Any type of objects have some common things, those are attributes (e.g., size, shape, color and weight) and behavior (e.g., a ball rolls, bounces; a baby cries, sleep and walks; a car accelerates, brakes and turns; a towel absorbs water).

→ All computer programs consist of two elements : code & data.

→ A program can be conceptually organized around its code (or) around its data.

→ That is, some programs are written around "what is happening" and others are written around "who is being affected".

→ These are the two paradigms (or) models that govern how a program is constructed.

→ The first model is called as process oriented model. The process oriented model can be thought of as code acting on data. Ex: C.

→ The second model is called as object-oriented model. The object-oriented model organizes a program around its data. i.e data acting on code. Ex: Java.

⇒ A Way of viewing the World :-

Suppose I wish to send flowers to a friend who lives in a city many miles away. Let me call my friend Sandy.

Because of the distance, there is no possibility of my picking the flowers and carrying them to Sandy's door myself.

→ So, I go down to my local florist, tell him the variety and quantity of flowers I wish to send to Sandy's address. I can be assured the flowers will be delivered.

→ Let me emphasize that the mechanism I used to solve my problem was following:

- find an appropriate agent (namely, Ganesh) and
- pass to him a message containing my request.

→ It is the responsibility of Ganesh to satisfy my request. I don't need to know the method used by Ganesh to satisfy my request.

→ Ganesh delivers a slightly different message to another florist in Sandy's city. That florist in turn perhaps has a subordinate who makes the floral arrangement.

→ The florist then passes the flowers, along with yet another, message to a delivery person.., and so on.

→ Earlier, the florist in Sandy's city had obtained his flowers from a flower wholesaler who, in turn, had interactions with the flower growers, each of whom had to manage a team of gardeners.

→ our first obervation of object-oriented problem solving is that the solution to my problem required the help of many other individuals.

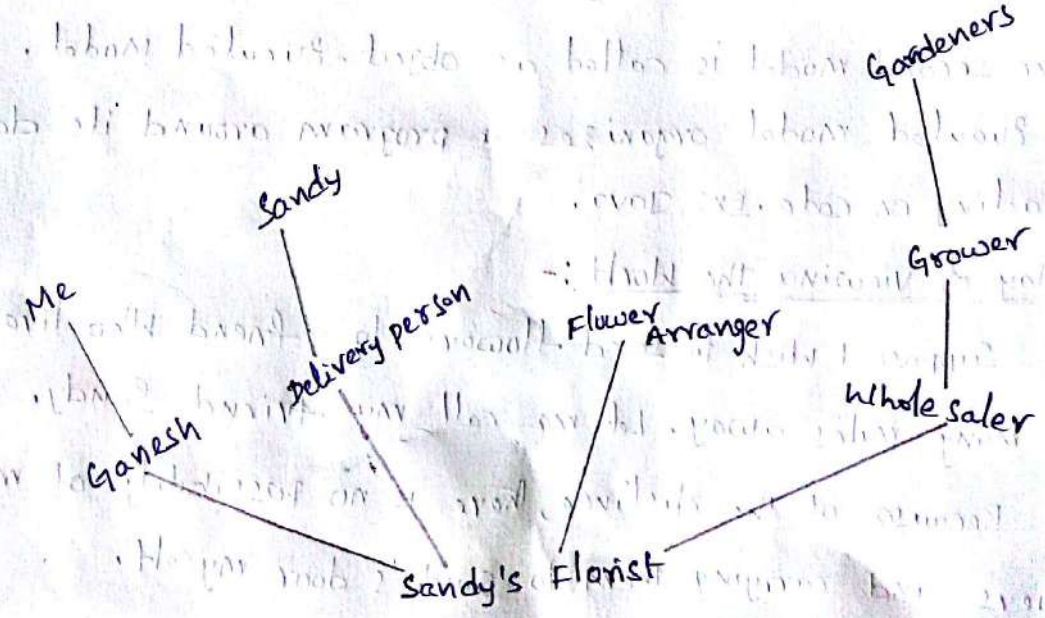


Fig: The community of agents helping me.

* Java OOPs Concepts :-

→ OOPs stands for object oriented programming System.

→ object means a real world entity such as pen, chair, table etc.

→ Object oriented programming is a methodology & paradigm to design a program using classes and objects.

→ It simplifies the software development and maintenance by providing some features & concepts.

→ The programming language where everything is represented as an object, is known as truly object-oriented programming language.

→ Smalltalk and Java are considered as the truely object-oriented programming language.

→ The following are the concepts (or) features (or) principals of object-oriented programming.

- Object
- Class
- Abstraction
- Encapsulation
- Inheritance
- polymorphism

• object :-

→ An object is a real world entity such as pen, chair, table, car, dog and etc.

→ Objects are key to understanding object-oriented technology.

In general, a real world objects have state and behavior. for example, a pen has state (color, company name, model) and behavior (writing, drawing). Ex: Motor bikes, dogs.

In software, the object's state is represented by variables and behavior is represented by methods.

Def^n:- "An object is a software bundle of variables and related Methods."

Example:- ① Object: car
State : color, make
Behavior: climb hill, SlowDown, Accelerate etc.

② Object : House
State : current location, color
Behavior : close/open main door.

• Class :-

A class is a collection of similar objects. In the real world, you often have many objects of the same kind. For example, your bicycle is just one of many bicycles in the world.

In terms of object-oriented, we say that your bicycle object is an instance of the class of objects known as bicycles.

Def^n:- " A class is a blueprint or prototype, that defines the variables and the methods common to all objects of a same kind.

Example: ① object : Byke
class : Bykes That same characterics.
② object : Dog • Each dog have same variables
class : Dogs and Methods like barking(), hungry()

• Abstraction :-

Abstraction is the concept of hiding the internal details and describing things in simple terms. For example: phone call, we don't know the internal processing.

Def^n:- "Hiding internal details and showing functionality is known as abstraction". Let us take the example of a car, we know that if accelerator pressed, speed will increase but don't know the internal process how speed will be increased.

• Encapsulation :-

Encapsulation is the technique used to implement abstraction in object-oriented programming.

Def^n:- "Binding code and data together into a single unit is known as encapsulation"

For example, capsule, it is wrapped with different medicines.

→ A Java class is the example of encapsulation.

→ In simple words, Encapsulation is a process of wrapping code and data into single unit. Let us take an example of a HR in a company. We communicate through HR not directly with the departments. HR is acting as public interface here.

• Inheritance :-

The process by which one class acquires the properties and functionalities of another class is known as inheritance.

Def^n:- "When one object acquires all the properties and behaviors of another (parent) object is known as inheritance".

→ It provides code reusability. It is used to achieve runtime polymorphism.

For example, A child inherits the properties of its parent.

Example: In object-oriented terminology, Mountain bikes, racing bikes and tandem bikes are all sub classes of the Bicycle super class.

→ Each subclass inherits the properties and functionalities of super class 'Bicycle'. (eg: speed, cadence, braking ...).

- Polymorphism:-

polymorphism is the concept where an object behaves differently in different situations. There are two types of polymorphism - compile time and runtime polymorphism.

Defⁿ:- "When one task is performed by different ways is known as polymorphism".

For example, To draw something e.g. shape a rectangle etc..

Example: The same message 'Move', the man walks, fish swim and birds fly.

* What is Java:-

Java is a programming language and a platform.

→ Java is a high level, robust, secured and object-oriented programming language.

→ Java is a high Level modern programming language, And it was introduced by "Sun MicroSystems' in 1995. It was developed by a "team under James Gosling".

→ platform is nothing but any hardware or software environment in which a program runs. Since Java has its own runtime environment i.e JRE(Java Runtime Environment)

* Where Java is used?

According to Sun, 3 billion devices run Java. There are many type of applications that can be created using Java programming. Some of them are as follows:

- Standalone Application:

It is also known as desktop application (or) window based application. An application that we need to install on every machine such as media player, antivirus and etc.

→ AWT and swing are used in Java for creating this Appln.

- Web Application:

An application that runs on the server side and creates dynamic page, is called Web application. Eg: irctc.coin

→ Servelt, JSP, Struts, jsf technologies are used for creating Web applications in Java.

- Enterprise Application:

An application that is distributed in nature, such as banking applications etc. "EJB" is used for this application

• Mobile Application :

An application that is created for mobiles.

→ currently Android and Java ME are used for creating Mobile applications.

* History of Java :-

→ Java team members (James Gosling, Mike Sheridan and patrick Naughton), initiated the Java language project in June 1991 for digital devices such as set-top boxes, televisions etc.

→ The small team of sun engineers called as "Green Team"

→ Firstly, it was called "Greentalk" by James Gosling and File extension was ".gt".

→ After that, it was called "oak". Why Oak? Oak is a symbol of strength and choosen as a national tree of many countries like U.S.A, France, Germany and etc.

→ In 1995, Oak was renamed as "Java" because it was already a trademark by Oak Technologies.

→ The team gathered to choose a new name. The suggested words were "dynamic", "revolutionary", "silk", "Jolt", "DNA" etc.

→ According to James Gosling "Java was one of the top choices along with silk". since Java was so unique, most of team members preferred Java.

→ Java is an island of Indonesia where first coffee was produced (called Java coffee).

→ Notice that Java is just a name.

→ originally developed by James Gosling at sun Microsystems and released in 1995.

* Features of Java (or) Java buzzwords :-

There are many features of Java. They are also known as Java buzzwords.

The features of Java given below

- Simple
- object-oriented
- platform independent
- Secured
- Robust
- Architecture neutral
- portable
- Dynamic
- Interpreted
- High performance
- Multithreaded
- Distributed

• Simple:

According to Sun, Java language is Simple because

→ Syntax is based on C and C++.

→ removed many confusing and rarely-used features e.g, Explicit pointers, operator overloading etc.

→ No need to remove unreferenced objects because there is automatic garbage collection in Java.

→ It eliminates the complexities of C and C++, therefore Java has been made simple.

• Object-Oriented:

→ object-oriented means we organize our software as a combination of different types of objects that contains both data and behaviour.

→ object-oriented programming (OOPs) is a Methodology that simplify software development and maintenance by providing some concepts & rules.

→ The basic concepts of OOPs are:

* Object
* class
* Inheritance
* polymorphism
* Abstraction
* Encapsulation

- **platform independent:**

→ A platform is the hardware or software environment in which a program runs.

→ There are two types of platforms: software-based and hardware-based. Java provides software-based platform.

→ Java code can be run on multiple platforms e.g. Windows, Linux, Mac os and etc.

→ Java code is compiled by the compiler and converted into byte code. This byte code is a platform independent code.

→ It is achieved by JVM (Java virtual Machine). The philosophy of Java is "Write Once, Run Anywhere (WORA).

- **Secured:**

→ Java is secured because Java does not use explict pointers and All Java programs runs inside the virtual machine sandbox.

→ Java uses the public key encryption system for providing security.

- **Robust:**

→ Robust simply means strong. Java is robust programming language because

* Java uses strong Memory Management.
* There are lack of pointers that avoids security problem.
* There is automatic garbage collection in Java.
* There is exception handling and type checking Mechanism in Java.

→ All these points makes Java robust.

- Architecture - neutral:

→ There is no implementation dependent features e.g. size of primitive types is fixed.

→ In 'C' programming, int data type occupies 2 bytes of memory for 32-bit architecture and 4 bytes of memory for 64-bit architecture. But in 'Java', it occupies 4 bytes of Memory for both 32 and 64 bit architectures.

- Portable:

→ Java is portable because we may carry the Java bytecode to any platform.

→ Java compiler is written in ANSI C with clean portability boundary.

→ The Java programs can run on any hard -ware environment.

- Dynamic:

→ Java is considered to be more dynamic than C or C++ since it is designed to adapt to an evolving environment.

→ Java programs can carry extensive amount of run-time information that can be used to verify and resolve accesses to objects on run-time.

- Interpreted:

→ Java byte code is translated on the fly to native machine instructions and is not stored anywhere.

→ Java byte code can be interpreted on any system that provides a Java Virtual Machine (JVM).

• High performance:

→ With the use of Just-In-Time compilers, Java enables high performance.

→ Java is faster than traditional interpretation.

→ Just-In-Time (JIT) compiler translates Java byte code directly into native machine code for very high speed performance.

• Multithreaded:

→ Java was designed to meet the real-world requirements. To accomplish this, Java supports multi-threaded programming, which allows you to write programs that do many things simultaneously.

→ A thread is like a separate program, executing concurrently.

→ The main advantage of multi-threading is that it doesn't occupy memory for each thread, it shares a common memory area.

• Distributed:

→ Java is designed for the distributed environment of the internet. We can create distributed applications in Java.

→ We may access files by calling the methods from any machine on the internet.

→ Java's remote method invocation (RMI) make distributed programs possible.