**Chapter 1**

**Introduction to Object Oriented Programming concepts**

**Learning Objectives**

After studying this chapter, you will be able to do:

* Procedure Oriented Programming
* Introduction to OOPs
* Principles of Object-Oriented Programming
* Difference between Procedure Oriented and Object-oriented Programming
* Introduction to JAVA
* Types of java programs – Applets and Applications
* Java Compilation process, Java Source code, Byte
* code, Object code, Java Virtual Machine (JVM)
* Features of JAVA

**Introduction**

Every programming language follows a certain approach to design the software. For a software to be implemented successfully , it is vital to make it as real as possible. Real life problems are very complex and it is important to convert these requirements into a model which can be developed using a programming language. You have many programming approaches like object oriented, procedure oriented, functional programming and many more. Let’s take a look at the differences between object oriented and procedure-oriented programming.

**Definition**

A set of instructions given to a computer to do a particular task is called program.

**Extra time**

Programming paradigm is an approach to solve a given problem by writing a set an instruction in a programming language. It refers to the organizing principles of a program.

**Procedure Oriented Programming**

A program in a procedural language is a list of instruction where each statement tells the computer to do something. It focuses on procedure (function) & algorithm is needed to perform the derived computation.

When program become larger, it is divided into function & each function has clearly defined purpose. Dividing the program into functions & module is one of the cornerstones of structured programming.

Example of POP are C, VB, FORTRAN, Pascal, etc.

**Characteristics of Procedure Oriented Programming:**

The characteristics of Procedure Oriented Programming are:

1. It focuses on process rather than data.

2. It takes a problem as a sequence of things to be done such as reading, calculating and printing. Hence, a number of functions are written to solve a problem.

3. A program is divided into a number of functions and each function has clearly defined purpose.

4. Most of the functions share global data.

5. Data moves openly around the system from function to function

**Limitations of Procedure Oriented Programming:**

The Limitations of Procedure Oriented Programming are:

1. Its emphasis on doing things. Data is given a second-class status even through data is the reason for the existence of the program.

2. Since every function has complete access to the global variables, the new programmer can corrupt the data accidentally by creating function. Similarly, if new data is to be added, all the function needed to be modified to access the data.

3. It is often difficult to design because the components function and data structure do not model the real world.

4. It is difficult to create new data types. The ability to create the new data type of its own is called extensibility. Structured programming languages are not extensible.

To overcome these limitations, Object Oriented programming was introduced.

**Object-Oriented Programming**

Object-Oriented Programming (OOP) is a programming paradigm centered around the concept of "objects," which are instances of classes that encapsulate data and behaviour. At its core, OOP emphasizes the organization of code into reusable, modular components that model real-world entities.

The examples of OOPs are Python, Java, Basic programming, and C++, etc

The key concepts of OOP are:

1. **Abstraction:** Abstraction involves simplifying complex systems by focusing on the essential characteristics while hiding unnecessary details. In OOP, abstraction is achieved through the use of abstract classes and interfaces. Abstract classes provide a blueprint for other classes but cannot be instantiated themselves, while interfaces define a contract for classes to implement certain methods without specifying the implementation details.

It is also known as data hiding within a program.

**Definition**

Abstraction means to represent the essential feature without detailing the background implementation or internal working detail.

**Example:**

An ATM abstracts the banking process. Users interact with the ATM interface to perform tasks like withdrawing cash, checking account balances, or transferring funds without needing to understand the inner workings of banking systems.

**Activity**

Try to think more example of abstraction in your daily life.

1. **Encapsulation:** Encapsulation is the bundling of data and methods that operate on that data within a single unit, typically a class. It allows for the hiding of implementation details from the outside world and enables better control over access to the data. This helps in creating more modular and maintainable code.

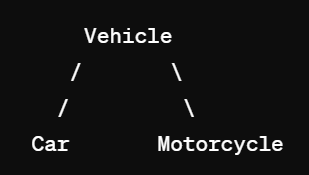
Example: Consider a car. The engine of the car encapsulates various internal components such as pistons, crankshafts, and camshafts. These components are hidden from the outside and interact with the external world only through well-defined interfaces like the accelerator pedal or ignition key.

Another example is a simple Soft Drink Vending machine. It serves you the drinks whenever you request. In this case, you do not know the background of how it serves you the drink when you press a button. This is Abstraction. It contains both data (soft drinks) and functions (serving drinks). This is called Encapsulation.

1. **Inheritance:** Inheritance is a mechanism by which a class can inherit properties and behavior from another class, known as the superclass or base class. The class inheriting from the superclass is called a subclass or derived class. Inheritance promotes code reusability and allows for the creation of hierarchical relationships between classes. For example, a "Vehicle" class may be a superclass, with subclasses like "Car" and "Truck" inheriting from it.

Example: Think about vehicles. You have a hierarchy of vehicles such as cars, trucks, and motorcycles. Each type of vehicle inherits common characteristics from a superclass "Vehicle," such as having wheels, an engine, and the ability to move. However, each subclass may also have its own unique attributes and behaviors. For instance, a truck might inherit from the Vehicle class but also have additional features like a larger cargo capacity.

Consider a scenario where we have a superclass called Vehicle, which represents common characteristics of vehicles. We also have two subclasses: Car and Motorcycle, which inherit from the Vehicle class.



In this diagram:

Vehicle is the superclass, which contains general attributes and behaviors shared by all vehicles.

Car and Motorcycle are subclasses of Vehicle, which inherit attributes and behaviors from the superclass while also having their own unique attributes and behaviors.

**Know More**

• The superclass is the class from which properties and behaviors are inherited.

• The subclass is the class that inherits properties and behaviors from the superclass.

1. **Polymorphism:** Polymorphism comes from the Greek words “poly” and “morphism”. “poly” means many and “morphism” means form i.e. many forms. Polymorphism allows objects of different classes to be treated as objects of a common superclass. It enables the same method to behave differently based on the object it is called on. Polymorphism can be achieved through method overriding (where a subclass provides a specific implementation of a method inherited from its superclass) and method overloading (where multiple methods with the same name but different parameters exist in the same class or hierarchy).

Example: In the animal kingdom, various species exhibit different behaviors, such as eating, sleeping, and moving.

Polymorphic Behavior: Despite the diversity of species, many animals share common behaviors. For instance, carnivores, herbivores, and omnivores all eat, but they consume different types of food. Similarly, animals like birds, mammals, and fish all move, but they use different locomotion methods (e.g., flying, walking, swimming). By defining a common superclass (Animal) with methods like eat() and move(), and allowing subclasses to override these methods with species-specific behavior, the animal kingdom demonstrates polymorphic behavior.

By utilizing these principles, OOP enables the development of modular, maintainable, and scalable software systems, making it a widely used and essential paradigm in modern software development.

**Extra Time**

There are two main types of polymorphism: Compile-time polymorphism and Run-time polymorphism.

**Fact time**

Alan Kay invented OOP.

**Characteristics of object-oriented programming**

The characteristics of object-oriented programming are:

1. OOP follows a bottom-up approach.
2. The program resulting from object-oriented programming is a collection of objects. Each object has its own data and a set of operations.
3. OOP restricts the free movement of data and the functions that operate on it. It uses a data/information hiding technique that allows better control over data. We will learn about it in the subsequent sections.
4. A properly defined class can be reused, giving way to code reusability.
5. The concept of object-oriented programming models real-world entities very well.
6. Due to its object-oriented approach, it is extremely useful in solving complex problems.

**Limitation of object-oriented programming**

The limitation of object-oriented programming are

1. The size of the programs created using this approach may become larger than the programs written using procedure-oriented programming approach.
2. Software developed using this approach requires a substantial amount of pre-work and planning.
3. OOP code is difficult to understand if you do not have the corresponding class documentation
4. In certain scenarios, these programs can consume a large amount of memory.

**Difference between Procedure Oriented and Object oriented**

OOP takes a bottom-up approach in designing a program. POP follows a top-down approach.

Program is divided into objects depending on the problem. Program is divided into small chunks based on the functions.

Each object controls its own data. Each function contains different data.

Focuses on security of the data irrespective of the algorithm. Follows a systematic approach to solve the problem.

The main priority is data rather than functions in a program. Functions are more important than data in a program.

The functions of the objects are linked via message passing. Different parts of a program are interconnected via parameter passing.

Data hiding is possible in OOP. No easy way for data hiding.

Inheritance is allowed in OOP. No such concept of inheritance in POP

C++, Java Pascal, Fortran

In short, Procedure-Oriented Programming focuses on procedures or functions that operate on data, while Object-Oriented Programming emphasizes the organization of code around objects that encapsulate data and behavior. OOP provides features like encapsulation, inheritance, and polymorphism, which promote code reuse, modularity, and maintainability.

**Elementary concept of objects and classes**

The main objective of object-oriented programming is to simulate the product as close as possible to the real world. How does it achieve that? Well, the answer is class and objects. These act as the fundamental blocks of object-oriented programming. Let’s learn about them a little more.

**Class:**

A class or object is a way of combining all sorts of data that relates to a single thing in one place, and a way of associating functions with that data. (For most intents and purposes, classes and objects are the same thing)

▪ It is a template or blueprint from which objects are created.

▪ It is a logical entity. It can’t be physical.

For example, you can consider human as a class. A car can be a class.

So, let’s say God made a class knows as Humans. Now, when he created you, you became a part of that class. Similarly, your friends and family are also a part of that class. Everything in the real world is categorised into some main category. This main category is usually what you call classes in programming.

A car is a common real-world object that can be modeled using a class in Java.



Car (Class):

The "Car" class serves as a blueprint for creating car objects. It encapsulates both data (attributes) and behavior (methods) of cars.

Attributes (Fields):

make, model, year, color, mileage: These are attributes of the "Car" class that represent the state or characteristics of a car object. Each car object will have its own values for these attributes.

Behaviors (Methods):

start(), stop(), accelerate(), brake(): These are methods of the "Car" class that represent actions or behaviors that a car object can perform. Methods encapsulate the behavior of a car, such as starting the engine, stopping the car, accelerating, and braking.

**Objects**

An entity that is a part of a class is known as an object. An object has some behaviour and state. For example, you are a part of class Human. That makes you an object of Human class.

**Extra time**

Java is a software bundle of related state and behaviour.

An object is what physically fills up the class. A class is just a blueprint but an object is an actual implementation of that blueprint. For example, God created a blueprint “Humans” which said that a human should have two legs, two arms, two ears, one mouth and the ability to talk. This is class Human. But when he created you, you were the actual implementation or result of his design of Humans. So, you are an object of class Human. Similarly, your friends and family are also objects of class Human.

**Know More**

The term object means a combination of data and logic that represents some real time entity.

Let’s take another example.

• A car is a road vehicle, typically with four wheels, powered by an internal combustion engine and able to carry a small number of people.

• Now, this is a blueprint of a car. Thus, a car is a class.

• Now, Maruti, Hyundai and Ford made their own versions of car. EcoSport, i20 and Brezza all have different designs and features but at the end of the day, they are all categorised into cars.

• Thus, these different Ford and Hyundai cars are all objects of a parent class “Car”.

**Extra time**

Classes and objects provide a way to model and represent real-world entities in Java programs. They enable code reuse, modularity, and abstraction, making it easier to manage and maintain complex systems.

**Software objects vs real world objects**

A real-world object is a physic body, on the other hand a software object is an abstraction of a data entity used in a programming paradigm known as Object Oriented Programming.

A software object is a logical representation of the real-world object and not the complete replica of it. For example, an employee of an organisation. Its software object can have properties like name, address, job, salary and so on but it cannot include every detail about that employee like how many degrees his elbow can bend.

In simple words, a software object tries to project itself as close as possible to the real-world object. That is how a software is created.

A software object has three characteristics:

• **State**: Represents data (value) of an object.

• **Behaviour**: Represents the behaviour (functionality) of an object such as deposit, withdraw etc.

• **Identity**: Object identity is typically implemented via a unique ID. The value of the ID is not visible to the external user. But it is used internally by the JVM to identify each object uniquely.

For example, your name, skin color, age, height, weight are the features that define you and define your state. The tasks you perform like talking, walking, working is your behaviour. You have a unique ID number that identifies you uniquely all over the country.

This was a general example of how to use concepts of object-oriented programming for real world problems.

**Introduction to Java**

Java programming language was originally developed by Sun Microsystems which was initiated by James Gosling and released in 1995 as core component of Sun Microsystems' Java platform (Java 1.0 [J2SE]).

**History of Java**

The history of Java is an interesting journey that begins in the early 1990s. Here's a brief overview:

1. **Origins at Sun Microsystems:**

Java was developed by James Gosling, Mike Sheridan, and Patrick Naughton at Sun Microsystems (later acquired by Oracle Corporation) in the early 1990s.

The project was initially named "Oak" after an oak tree outside Gosling's office. It was later renamed "Java" due to trademark issues.

1. **Public Release (1995):**

Java 1.0, the first official version of Java, was released to the public by Sun Microsystems in 1995.

It included the Java Development Kit (JDK) and the Java Runtime Environment (JRE).

1. **Key Features:**

Java was designed with the goal of being platform-independent, secure, and robust.

It introduced features such as automatic memory management (garbage collection), exception handling, and a rich set of standard libraries.

1. "**Write Once, Run Anywhere":**

One of Java's most significant contributions to software development is its platform independence.

Java programs are compiled into bytecode, which can run on any device with a Java Virtual Machine (JVM), regardless of the underlying hardware or operating system. This principle is often summarized as "write once, run anywhere" (WORA).

1. **Popularity and Adoption:**

Java quickly gained popularity due to its simplicity, portability, and suitability for building a wide range of applications, from desktop to web and enterprise systems.

It became the language of choice for many developers and organizations, leading to widespread adoption in various industries.

1. **Evolution and Versions:**

Over the years, Java has continued to evolve with regular releases and updates introducing new features, enhancements, and improvements.

Major releases include Java 2 (renamed as Java SE), Java EE (Enterprise Edition) for enterprise applications, and Java ME (Micro Edition) for mobile and embedded devices.

Recent versions include Java 8, which introduced lambda expressions and the Stream API, and Java 11, which marked the transition to a faster release cadence with a new long-term support (LTS) model.

1. **Open Sourcing:**

In 2006, Sun Microsystems open-sourced the Java platform under the GNU General Public License (GPL) through the OpenJDK (Java Development Kit).

This move further contributed to Java's widespread adoption and allowed the community to contribute to its development.

1. **Acquisition by Oracle:**

In 2010, Oracle Corporation acquired Sun Microsystems, becoming the steward of the Java platform.

Oracle continues to develop and maintain Java, overseeing its evolution and providing updates and support to the community.

Throughout its history, Java has remained a versatile and powerful programming language, playing a significant role in shaping the modern software development landscape. It continues to be used extensively in a wide range of applications, from enterprise systems and web services to mobile apps and cloud computing.

**Fact Time**

Java was originally known as OAK.

**Translators**

The computer language that is the machine code is almost impossible to be understood by the humans. So, humans write their programs in high level languages and then this is converted to low level language so that the computer can understand it. So, who does this conversion? Here comes the Translators.

Translators are software’s which are used to convert high level languages to low level languages. There are two types of translators:

1. Interpreters
2. Compilers

**Know More**

The Java development kit comes with a collection of tools that are used for developing and running java programs.

| **Feature** | **Compiler** | **Translator** |
| --- | --- | --- |
| Purpose | Converts entire source code into machine code or intermediate code before execution. | Converts a program from one language to another without necessarily producing machine code. |
| Input | Takes the entire source code as input. | Takes the source code or program in one language as input. |
| Output | Produces machine code or intermediate code that can be directly executed by the computer. | Produces equivalent source code or program in another language. |
| Execution | Execution occurs after the entire source code is compiled. | Translation happens before the execution of the program. |
| Efficiency | Generally produces faster execution as the entire code is optimized before execution. | May introduce overhead due to translation process and may not be as efficient as natively compiled code. |
| Examples | C/C++ compilers like GCC, Clang. Java compiler (javac). | Language translators like Google Translate (translates text between languages), Java bytecode translator (translates Java bytecode to machine code during execution). |

There are two type of Java programs:

1. **Applications**: Java programs that run directly on your machine.

a. Applications must have a main().

b. Java applications are compiled using the javac command and run using the java command.

**Extra Time**

The term java application generally refers to an application that is designed for stand-alone use. A frequently used slang term for application is app.

2. **Applets**: Java programs that can run over the Internet. The standard client/server model is used when the Applet is executed. The server stores the Java Applet, which is sent to the client machine running the browser, where the Applet is then run.

a. Applets do not require a main(), but in general will have a paint().

b. An Applet also requires an HTML file before it can be executed.

c. Java Applets are also compiled using the javac command, but are are run either with a browser or with the appletviewer command.

**Know More:**

The term Java applet generally refers to a small application that is designed for use on the World Wide Web.

Before understanding the java compilation program, let us first understand some of the basic terms used in Java.

**Source code**: These are the original instructions written by a programmer in Java language following the rules of Java.

* Java source code refers to the human-readable text written by developers to create Java programs.
* Source code files typically have a .java extension.
* Java source code contains classes, methods, variables, and other elements of the Java programming language.

**Bytecode:** Bytecode is program code that has been compiled from source code into low-level code designed for a software interpreter. It may be executed by a virtual machine (such as a JVM) or further compiled into machine code, which is recognized by the processor.

* Bytecode is an intermediate representation of Java programs generated by the Java compiler (javac) during the compilation process.
* It is a set of instructions for the Java Virtual Machine (JVM) to execute.
* Bytecode is platform-independent and can be executed on any device or operating system with a compatible JVM implementation.
* Bytecode files have a .class extension and are stored in compiled form.

**Object code**: Object code is code generated by a compiler or other translator, consisting of machine code, byte code, or possibly both, combined with additional metadata that will enable a linker, loader, or linker-loader to assemble it with other object code modules into executable machine code or byte code.

• In the context of Java, object code is often associated with native code generated by a Just-In-Time (JIT) compiler or ahead-of-time (AOT) compiler.

* Object code refers to the machine-readable binary code generated by a compiler or an assembler from source code.

• Unlike bytecode, which is executed by the JVM, object code is executed directly by the underlying hardware.

• Object code is specific to the target platform and cannot be executed on different platforms without modification.

**JVM:** The Java Virtual Machine (JVM) is an abstract computing machine that enables Java bytecode to be executed on any device or operating system.

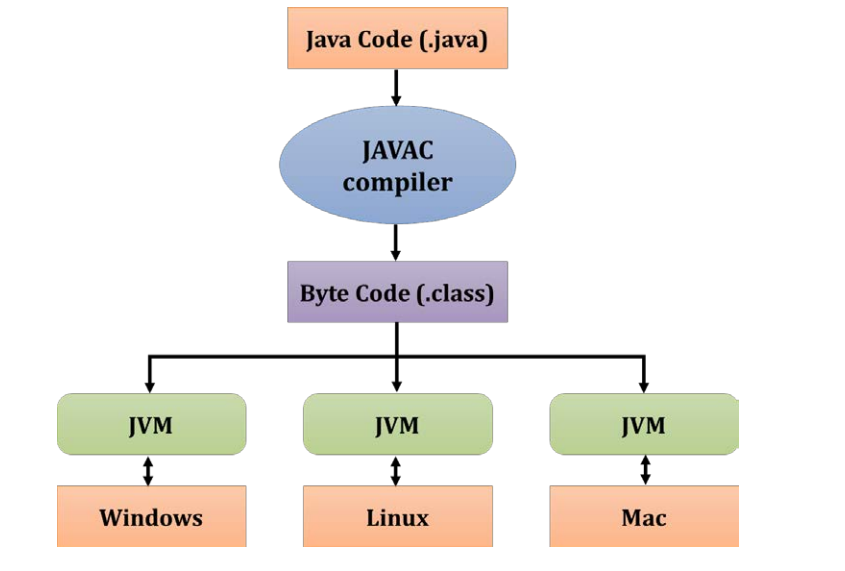
It acts as an interpreter for Java bytecode, translating it into machine-specific instructions that can be executed by the underlying hardware.

The JVM provides features like memory management, garbage collection, and security to Java programs.

It ensures that Java programs run efficiently and securely on a wide range of platforms.

**Let’s discuss what does JVM performs?**

It is used to load the code. It Verifies and executes the code. And it provides run time environment.



Here are some key points about the JVM:

1. **Platform Independence:** One of the main features of the JVM is its platform independence. Java programs are compiled into bytecode, which is a platform-neutral intermediate representation of the program. The JVM then interprets this bytecode and executes it on any device or operating system that has a JVM implementation.
2. **Execution Environment**: The JVM provides an execution environment for Java programs, including memory management, garbage collection, and security features. It ensures that Java programs run efficiently and securely on a wide range of platforms.
3. **Just-In-Time (JIT) Compilation**: In addition to interpreting bytecode, many modern JVM implementations also use a technique called Just-In-Time (JIT) compilation. This involves compiling bytecode into native machine code at runtime, which can improve performance by executing optimized native code instead of interpreting bytecode.
4. **Class Loading and Runtime Management**: The JVM is responsible for loading classes into memory as they are needed by the program. It also manages memory allocation and deallocation, including garbage collection to reclaim memory occupied by objects that are no longer in use.
5. **Security:** The JVM includes built-in security features to protect against malicious code. It enforces access control policies, verifies bytecode to ensure it does not violate security constraints, and provides a secure execution environment for Java programs.

**Extra Time**

The process of converting the source code into machine code is called compilation. The compilation produces the machine code. For different platforms different machine codes are produce.

**Java API (Java Application Programming Interface)**

Java API is a library of compiled code (small built-in programs) that can be used in our program in order to lessen our program effort.

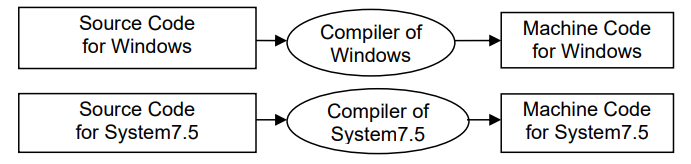
**Java Platform**

The Java Compiler, JVM combined with Java APIs makes Java platform. JDK (Java Development Kit): It contains Java compiler, Java Virtual Machine (JVM), Java API (libraries) etc.

**Java Compilation**

**Ordinary Compilation Process**

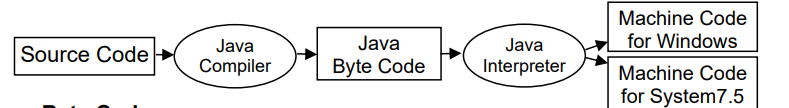
The compiler for Windows converts the source program directly to machine code for Windows only. The compiler for System7.5 compiles to machine code only for System7.5. The compiler for a particular system software can convert only for that system software.



**Java Compilation Process**

Java Compiler compiles the source code to Java Byte Code, then this byte code is interpreted by Java Interpreter into Machine Code for a specific platform i.e., if the operating system is Windows then to the machine code for Windows, if the operating system is System7.5 then to the machine code for System7.5

Diagrammatic Representation of Java Compilation Process



**Features of Java**

Java is:

1. **Object Oriented**: In Java, everything is an Object. Java can be easily extended since it is based on the Object model.
2. **Platform Independent**: Unlike many other programming languages including C and C++, when Java is compiled, it is not compiled into platform specific machine, rather into platform independent byte code. This byte code is distributed over the web and interpreted by the Virtual Machine (JVM) on whichever platform it is being run on.
3. **Simple:** Java is designed to be easy to learn. If you understand the basic concept of OOP Java, it would be easy to master.
4. **Secure:** With Java's secure feature it enables to develop virus-free, tamper-free systems. Authentication techniques are based on public-key encryption.
5. **Portable**: Being architecture-neutral and having no implementation dependent aspects of the specification makes Java portable. Compiler in Java is written in ANSI C with a clean portability boundary, which is a POSIX subset.
6. **Robust**: Java makes an effort to eliminate error prone situations by emphasizing mainly on compile time error checking and runtime checking.
7. **Multithreaded:** With Java's multithreaded feature it is possible to write programs that can perform many tasks simultaneously. This design feature allows the developers to construct interactive applications that can run smoothly.
8. **Interpreted:** Java byte code is translated on the fly to native machine instructions and is not stored anywhere. The development process is more rapid and analytical since the linking is an incremental and light-weight process.
9. **High Performance**: With the use of Just-In-Time compilers, Java enables high performance.

**Know More**

A package is a group of related classes and interfaces. It provides a convenient mechanism for managing a large set of classes and interfaces and avoiding naming conflicts. Some Java packages are java.io, java.util, java.awt, and java.net.

**Importance of Java**

1. **Java is Platform Independent:** The program written in Java can be run on any platform that means it can be executed by any operating system and any processors; because the source code is compiled by Java Compiler to byte code which is suited to any platform and then interpreted by JVM to native executable code. S,o Java is platform independent.
2. **Java is a Programming Language as well as a Platform:** Java can be used to develop various application programs like MS Word, PowerPoint, Excel etc. So, Java is a programming language. Java includes some programs like compiler, interpreter - generally which are parts of a system software. So, Java is a platform also.

**Fact time**

Java is owned by Oracle, and more than **3 billion**devices run Java.

**Check your skills**

**State whether the following statements are True or False.**

1. Java application is a Java program which is developed by users. True
2. James Genes developed Java programming language. False

**Keywords**

* **Class:** In Java, a class is a blueprint or template for creating objects.
* **Objects:** Objects are instances of classes.
* **Polymorphism**: Polymorphism is a concept in object-oriented programming that allows objects of different types to be treated as objects of a common superclass.
* **Encapsulation:** Encapsulation is the bundling of data (attributes) and methods (behaviors) that operate on that data into a single unit, typically a class
* **Java:** Java is a high-level programming language known for its platform independence, simplicity, and robustness.
* **Source Code:** Source code refers to the human-readable instructions written in a programming language, such as Java.
* **JVM (Java Virtual Machine):** JVM is an abstract computing machine that enables Java bytecode to be executed on any device or operating system.

**Recap**

• Object oriented programming is an approach which uses data more than functions. This data is called object.

• Procedure oriented programming put more emphasis on functions rather than data.

• There are 4 principles of object-oriented programming.

• Abstraction refers to the practice of showing only the relevant details and hiding the unimportant details.

• Encapsulation means binding the data and functions together in one solution.

• Inheritance is one of the most important characteristics of object-oriented languages where one module borrows the features of some other module and hence helps is reusing the same module at multiple places.

• Polymorphism is derived from two words- poly and morph. Poly means many and morph means forms. It refers to the concept of one thing doing multiple tasks.

• Java is an object-oriented language developed by Sun Microsystems.

• Source code is the original instructions generated by human programmer.

• Bytecode is program code that has been compiled from source code into low-level code designed for a software interpreter.

• Object code is the code output of compiler.

• A Java Virtual Machine (JVM) is a virtual machine that enables a computer to run Java programs as well as programs written in other languages and compiled to Java bytecode.

• Some features of Java include multithreading, portability, simplicity and robustness.

**Exercises**

**A. Multiple choice questions.**

1. In OOP, what is the process of bundling data and methods that operate on the data into a single unit known as?

a. Polymorphism

b. Inheritance

c. Encapsulation

d. Abstraction

Ans: c

2. Which OOP principle allows objects of different classes to be treated as objects of a common superclass?

a. Inheritance

b. Encapsulation

c. Polymorphism

d. Abstraction

Ans: a

3. What is the term for the ability of a method to have multiple forms of implementation?

a. Inheritance

b. Encapsulation

c. Polymorphism

d. Abstraction

Ans: c

4. Name the programming technique that specifies a series of well-structured steps and procedures within its programming context to compose a program.

a. Procedure Oriented Programming

b. Modular Programming

c. Object Oriented Programming

d. None of these

Ans: a

5. Which OOP principle focuses on hiding the internal implementation details of a class from the outside world?

a. Inheritance

b. Encapsulation

c. Polymorphism

d. Abstraction

Ans: b

6. Which OOP principle promotes code reuse by allowing a class to inherit attributes and methods from another class?

a. Inheritance

b. Encapsulation

c. Polymorphism

d. Abstraction

Ans: a

7. Name the programming technique that implements programs as an organized collection of interactive objects.

a. Procedure Oriented Programming

b. Modular Programming

c. Object Oriented Programming

d. None of these

Ans: c

8. What is the primary benefit of using OOP concepts in software development?

a. Faster execution speed

b. Easier debugging

c. Better code organization and reusability

d. Lower memory usage

Ans: c

9. A set of instructions given to a computer to do a particular task.

a. Program

b. High level Language

c. Object

d. None of these

Ans: a

1. An object is represented by two attributes, out of which one is characteristics and the other one is \_\_\_\_\_\_\_\_\_\_\_.

a. Behaviour

b. Situation

c. Abstraction

d. Encapsulation

Ans: a

1. Constructor overloading follows which principle of Object-Oriented programming?
2. Inheritance
3. Polymorphism
4. Abstraction
5. Encapsulation

Ans: b

12. Java is a \_\_\_\_\_\_\_ programming language.

a. Compiled

b. Interpreted

c. Both compiled and interpreted

d. None of the above

Ans: c

13. Which of the following is NOT a feature of Java?

a. Platform independence

b. Pointers

c. Object-oriented

d. Robustness

Ans: b

14. Java programs are compiled into \_\_\_\_\_\_\_ code.

a. Source

b. Machine

c. Bytecode

d. Binary

Ans: c

**B. Fill in the blanks.**

1. Java is a \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_language. (Case-Sensitive)

2. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**developed Java programming language. (James Gosling)**

**3. Java compiler converts Java source code into an intermediate binary code called\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. ( Bytecode)**

**4. The full form of JVM is \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. (Java Virtual Machine)**

**5. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**translates the source program into target program one line at a time. (Interpreter)

6. The core program or text which is written in a language like Java is called \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. (Source Program)

7. The process of combining data and functions that are used together as a unit is called \_\_\_\_\_\_\_\_\_. (Encapsulation)

8. A \_\_\_\_\_\_\_\_is a blueprint that defines data and functions common to all the objects. (Class)

9. Syntax of Java is based on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. (C++)

10. Java is a software bundle of related state and \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. (behaviour).

**C. State whether the following statements are True or False.**

1. Java is a procedural programming. F

2. Wrapping of data and functions together as a single unit is known as encapsulation. T

3. Polymorphism comes from the Latin words. F

4. Class is the process by which one object can acquire the properties of another. F

5. Alan Kay invented OOP. T

6. Polymorphism allows for class-specific behavior. T

7. Applet programs are meant to run in a web page on the Internet. T

8. Java Servlets programs run in computers that provide web services. T

9. Java originally developed by Sun Microsystems and released in 1994. F

10. In Java, there are two types of programs. T

D. **Very short answer type questions.**

1. Expand the following terms: JDK, JVM, and IDE.

Ans. JDK: Java Development Kit JVM: Java Virtual Machine IDE: Integrated Development Environment

2. What was the earlier name of Java?

Ans. The earlier name of Java was Oak.

1. Who developed Java?

Ans. Java was developed by James Gosling.

1. Define source code.

Ans. Source code is a set of instructions and statements written by a programmer using a computer programming language.

1. Define objects. Also, give a real-life example.

Ans. An object is a unique entity, which has some characteristics and behaviours. For example, a student can be considered as an object because he/she has few characteristics like name, class, age, etc.

1. Why is Java referred to as platform-independent?

Ans. Java is referred to as platform-independent because the java compiled code (byte code) can run on all operating systems.

**E. Short questions.**

1. How is Java platform independent?

Ans: Java achieves platform independence by compiling source code into bytecode, which can run on any system with a Java Virtual Machine (JVM). The JVM interprets bytecode into machine code specific to the underlying hardware and operating system. Java's standard libraries abstract away platform-specific details, ensuring consistency across different environments. This allows Java programs to be developed and executed seamlessly across various platforms without modification.

2. Describe the Java compilation process.

Ans:

1. Java source code (.java) is compiled by the Java compiler (javac) into bytecode (.class) files.
2. Bytecode is platform-independent and can be executed on any system with a Java Virtual Machine (JVM).
3. At runtime, the JVM loads and interprets bytecode, or it may use Just-In-Time (JIT) compilation to translate bytecode into native machine code for improved performance.
4. The JVM interacts with the underlying operating system to execute the program, utilizing platform-specific resources as needed.
5. This process ensures that Java programs can be developed once and run on any platform with a compatible JVM, enabling platform independence.
6. Define the term Byte Code.

Ans. The Java compiler compiles the source programs into an intermediate code called the Java Byte Code which is interpreted by the Java Virtual Machine (JVM)

1. What is inheritance and how it is useful in Java. Give examples. [2008]

Ans: It is process by which objects of one class acquire the properties of objects of another class. Inheritance supports the concepts of hierarchical representation. In OOP the concepts of inheritance provides the idea of reusability. e.g. A class CAR inherits some property from the class Automobiles, which in turn inherits properties from Vehicle class. The capability to pass down properties and so allows us to describe things efficiently.

1. What role does polymorphism play as Java feature?

Ans: It means the ability for a message or data to take more than one form. For example, an operation, many types of data used in the operation. E.g. a child behaves like a student in school, customer in a shopping mall, passenger in a bus and like a son/daughter at home. Same person behave differently in various circumstances.

1. **What does a class encapsulate?**

**Ans:** A class encapsulated the data (instance variables) and methods.

1. Explain the two types of Java programs.

Ans: There are two type of Java programs:

a. Applications: Java programs that run directly on the machine.

b. Applets: Java programs that can run over the Internet. The standard client/server model is used when the Applet is executed. The server stores the Java Applet, which is sent to the client machine running the browser, where the Applet is then run.

1. **State the Java concept that is implemented through:**
2. **a super class and a subclass.**
3. **the act of representing essential features without including background details.**

**Ans**: a. Inheritance

b. Abstraction

1. What are the features of Java?

Ans: The features of Java as follows:

1. It is an object-oriented language, simple and robust.
2. Java program is both compiled and interpreted.
3. Write once run anywhere (WORA). The java programs are written once, but can be run on different platform without making changes.
4. Has enhanced security features.
5. Platform independent.
6. Write the differences between Procedural Programming and Object-Oriented Programming.

Ans: Procedural programming aims more at procedures. The emphasis is a doing thing rather than the data being used. In procedural Programming paradigm data are shared among all the functions participating thereby risking data safety and security. Object Oriented Programming is based on principles of data hiding, abstraction, inheritance and polymorphism. It implements programs using classes and objects, In OOP‟s data and procedure both given equal importance. Data and functions are encapsulated to ensure data safety and security.

1. What is the use of java API?

Ans: Java API (Application Programming Interface) serves as a comprehensive library of classes, interfaces, and methods that developers utilize to build Java applications. It offers ready-made solutions for common tasks like file I/O, networking, database connectivity, GUI development, and more. By providing standardized interfaces, it ensures portability and interoperability across different Java implementations. Java API encourages code reuse, promoting efficiency and consistency in software development. It abstracts away low-level details, allowing developers to focus on higher-level logic and functionality. Ultimately, Java API facilitates rapid development, enhances maintainability, and enables the creation of robust and scalable Java applications.

1. Class is termed as an object factory. Explain.

Ans: A class is known as an object factory because it contains all the statements needed to create an object, its attributes as well as the statements to describe the operations that the object will be able to perform.

1. What is the difference between an object and a class?

Ans. Object Class

1. Object is an instance of a class. 1. Class is a blueprint or template from which objects are created.

2. Object is a real-world entity. 2. Class is a group of similar objects.

3. Object is a physical entity. 3. Class is a logical entity.

4. Object is created mainly through new keyword. 4. Class is declared using class keyword.

1. Explain the element of OOP which allows the properties of one class to be used by another class.

Ans: In Java, this feature can be obtained using inheritance. It is the process by which objects of one class acquire the properties of objects of another class. Inheritance supports the concept of hierarchical representation. In OOP, the concept of inheritance provides the idea of reusability.

1. What is meant by a base class and a derived class?

Ans: the class which other classes are derived through the process of inheritance is called base class or superclass.

The class that inherits properties from a base class is called a derived class.

**F. Application based questions.**

1. Give a real-life example to explain the term encapsulation.

Ans: A capsule is a real-life example of encapsulation. Basically, a capsule encapsulates several combinations of medicine. If combinations of medicine are variables and methods, then the capsule will act as a class and the whole process is called encapsulation.

2. Explain the concept of class and objects by providing a real-life example.

Ans: A class is a group of objects that have some common characteristics and behaviours. A class is a blueprint from which individual objects are created. In other words, a class is a concept, and the object is the implementation of that concept. We need to have a class before creating the objects. Let us consider a car as a class that has characteristics, like steering wheel, seats, brakes, etc., and its behaviour is mobility. However, you can say BMW, having a registered number of 4284, is an 'object’ that belongs to the class 'car'.

**Unsolved Exercise**

1. Which of the following is a procedure-oriented programming language?

a. Pascal

b. C++

c. Java

d. Python

Ans: a

2. Name the object-oriented language among the following.

a. Pascal

b. Fortran

c. C++

d. All of these

Ans: c

3. What is the most important entity in object-oriented programming languages?

a. Objects

b. Functions

c. Variables

d. Expressions

Ans: a

4. Name the concept of showing only important details and hiding non-essential details.

a. Abstraction

b. Encapsulation

c. Inheritance

d. Polymorphism

Ans: a

5. What is used to run bytecode?

a. JVM

b. JPM

c. JRE

d. JDK

Ans: a

6. Name the java programs that run directly on a machine.

a. Applets

b. Applications

c. Programs

d. Websites

Ans: b

7. What are the type of java programs that run on the Internet?

a. Applets

b. Applications

c. Programs

d. Websites

Ans: a

8. Name the characteristic of Java to run on any platform.

a. Robust

b. Portability

c. Simple

d. Multithreaded

Ans: b

9. What is the feature of Java to run parallel tasks at the same time called?

a. Robust

b. Portability

c. Simple

d. Multithreaded

Ans: d

10. Java was originally known as:

1. OAK
2. ROSE
3. WALNUT
4. All of these

Ans: a

B. **State whether the following statements are True or False.**

1. Java is a procedure-oriented language. F

2. An object is a real- world entity. T

3. Pascal is a procedure- oriented language. T

4. Byte code is the original code written by a programmer. F

5. JVM stands for Java Virtual Machine. T

6. Object code is the output of a compiler. T

7. Java supports multithreading. T

8. Java is a platform dependent language. f

9. Sun Microsystems developed Java. T

10. JVM is java compiler. F

**C. Fill in the blanks.**

1. \_\_\_\_\_\_\_\_\_\_\_\_\_was the first purely object-oriented programming language. (SmallTalk)

2. OOP first came into picture in \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ by Alan and his team.( 1970’s)

3. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ indicates code reusability in OOPs. (Inheritance)

4. Java is a platform\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. (Independent)

5. The full form of JIT is \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. (Just in Time)

6. Java program comes into \_\_\_\_\_\_\_\_\_\_\_\_\_main types. (2)

7. The concept of deriving one class from another class is called\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. (Inheritance)

8. The Java compiler translates Java source code into Java\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. (Byte code)

9. A \_\_\_\_\_\_\_\_\_\_\_\_\_is a program that behaves like a computer. (JVM)

10. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_is a high-level programming language. (Java)

**D. Answers the following questions.**

1. Name some of the different programming approaches used these days.

Ans: Several programming approaches are used in modern software development. Some of them include:

1. Procedural Programming: Focuses on writing procedures or routines to perform specific tasks, often using a linear sequence of steps.
2. Object-Oriented Programming (OOP): Organizes code into objects, which encapsulate data and behavior, promoting modularity, reusability, and maintainability.
3. How is encapsulation different from abstraction?

Ans: Encapsulation is about bundling data and methods into a single unit and controlling access to them, whereas abstraction is about defining clear interfaces and hiding unnecessary implementation details. Encapsulation is a mechanism for implementing abstraction, but abstraction is a broader concept that encompasses encapsulation along with other principles like inheritance and polymorphism

1. Differentiate source code, bytecode and object code.

Ans: Source code is the human-readable form of a program, written in a programming language like Java or C++. Bytecode is an intermediate representation generated by compiling the source code, specific to a virtual machine like the JVM. Object code, also known as machine code, is the binary executable form of a program, directly executable by the computer's hardware. While source code is portable, bytecode requires a compatible virtual machine, and object code is platform-specific.

1. What do you understand by data abstraction? Explain with an example.

Ans. Abstraction refers to representing the essential features of a system without considering all the details. Example: When we drive a car, we concentrate on how to drive it without bothering ourselves on how the engine works and other thing.

5. Describe four features of Java.

1. Robust: Java is a robust and reliable programming language. Its powerful compiler checks for early errors during compilation and the interpreter checks for any run time errors thus, eliminating situations that are error-prone.

2. Secure: Java is secure because it allows applets to confine to the java runtime environment only. Applets cannot access any other part of the computer when downloaded one the web browser. This helps to eliminate any damage due to malicious software. For example: accessing passwords, or finding you bank details by searching the contents of your computer.

3. Object Oriented: Java is a rue object-oriented programming language because it threats everything as an object. This enable us to take full advantage of the object-oriented paradigm and its capabilities or creating flexible and modular programs.

4. Multithreaded: Java is also a multithreaded programming language. Multithreading works in a similar way a multiple process run on one computer. To understand it, let’s take the example of a Java game. Although the game is a single program, yet different threads within the game manage scores, graphics and hundreds of the core features of the game, hence, a single application executes multiple threads together which enhances the capabilities of a program.

6. What are the characteristics of procedural programming?

Ans: The characteristics of procedural programming are:

I. Procedural programming follows a top-down approach.

II. The program is divided into blocks of codes called functions, where each function performs a specific task.

III. Procedural programs model real-world processes as 'procedures' operating on 'data'.

IV. The data and function are detached from each other.

V. The data moves freely in a program. vi. It is easy to follow the logic of a program.

VI. A function can access other function's data by calling that function.

7. What are the limitation of procedural programming?

Ans: The limitation of procedural programming are:

1. Procedural programming mainly focuses on procedures or functions. Less attention is given to the data.
2. The data and functions are separate from each other.
3. Global data is freely moving and is shared among various functions. Thus, it becomes difficult for programmers to identify and fix issues in a program that originate due to incorrect data handling.
4. Changes in data types need to be carried out manually all over the program and in the functions using the same data type.
5. Limited and difficult code reusability. VI. It does not model real-world entities (e.g., car, table, bank account, loan) very well where we, as a human being, perceive everything as an object.

8. What are the characteristics of object-oriented programming?

Ans: The characteristics of object-oriented programming are:

I. OOP follows a bottom-up approach.

II. The program resulting from object-oriented programming is a collection of objects. Each object has its own data and a set of operations.

III. OOP restricts the free movement of data and the functions that operate on it. It uses a data/information hiding technique that allows better control over data. We will learn about it in the subsequent sections.

IV. A properly defined class can be reused, giving way to code reusability.

V. The concept of object-oriented programming models real-world entities very well.

VI. Due to its object-oriented approach, it is extremely useful in solving complex problems.

1. What are the limitation of object-oriented programming?

Ans: The limitation of object-oriented programming are:

I. The size of the programs created using this approach may become larger than the programs written using procedure-oriented programming approach.

II. Software developed using this approach requires a substantial amount of pre-work and planning.

III. OOP code is difficult to understand if you do not have the corresponding class documentation

IV. In certain scenarios, these programs can consume a large amount of memory.

1. Java is a case sensitive. Explain.

Ans: Java is case sensitive means that it distinguishes between upper case and lower-case characters.

1. Explain basic principles of Java.

Ans: Basic Principles of OOP:

1. Abstraction: The act of representing essential features, without including the background details.

2. Inheritance: Capability of one class of things to inherit capabilities or properties from another class.

3. Encapsulation: Wrapping up of data and functions into a single unit.

4. Polymorphism: Polymorphism is the ability for a message or data to be processed in more than one form.

1. How does ordinary compilation take place?

Ans. The process of converting the source code into machine code is called compilation. The compilation produces the machine code. For different platforms different machine codes are produce.

1. "Java uses both a compiler and an interpreter." Explain the statement.

Ans: In java compilation, first, the compiler converts the source code to an intermediate language (Java Byte Code), which is further converted to machine code by the interpreter (Java Virtual Machine)

1. Why is Java called a robust language?

Ans: Java is called a robust language because it is capable of handling run-time errors, supports automatic garbage collection and exception handling, and avoids explicit pointer concept. Java has a strong memory management system. It helps in eliminating errors as it checks the code during both compile and runtime.

1. Give an example to explain data abstraction. (ICSE 2018)

Ans: Consider a car. When you drive a car, you don't need to understand all the intricate details of how the engine works or how the transmission system functions. Instead, you interact with the car at a higher level of abstraction. You have a steering wheel to control the direction, pedals for acceleration and braking, and indicators to show your speed and fuel level. These are the essential interfaces or methods for interacting with the car.

Now, let's relate this to data abstraction in programming. Imagine you're developing a software application for a car rental service. Instead of needing to understand every detail of how the car's engine works, you create an abstract representation of a car in your program. This abstract representation includes essential properties and behaviors of a car that are relevant to your application, such as the model, color, fuel level, and methods to drive, park, and refuel the car.

By using data abstraction, you hide the complex inner workings of the car (like the engine, transmission, etc.) and expose only the relevant details and functionalities that users of your program need to interact with. This simplifies the programming process, enhances code readability, and allows for easier maintenance and scalability of the software application.

**Lab Work**

1. What type of programming approach is used by C++?
2. Try to find about the location where a byte code is generated.
3. Compile one Java program.

**Project work**

1. Find out at least two other languages other than C++ and Java which are based on object-oriented approach.
2. Is Java an interpreted language or a compiled language?
3. Find out the usage of multithreading in real world scenarios.

**Teacher’s Notes**

* Help the students understand the significance of object-oriented programming
* Help them understand the different features of Java