**Core Java**

**Q1 -> What do you understand by pure object-oriented programming language? Why java is not a pure object-oriented programming language?**

The programming language is called pure object-oriented language that treats everything inside the program as an object. The primitive types are not supported by the pure OOPs language. There are some other features that must satisfy by a pure object-oriented language

* Encapsulation
* Inheritance
* Polymorphism
* Abstraction
* All predefined types are objects
* All user-defined types are objects
* All operations performed on objects must be only through methods exposed to the objects.

**Java is not a pure object-oriented programming language** because pre-defined data types in Java are not treated as objects. Hence, it is not an object-oriented language

## Q2 -> What is Exception Handling in Java?

## In Java, an exception is an event that disrupts the normal flow of the program. It is an object which is thrown at runtime.

## Exception Handling is a mechanism to handle runtime errors such as ClassNotFoundException, IOException, SQLException, RemoteException, etc

### **Advantage of Exception Handling**

The core advantage of exception handling is **to maintain the normal flow of the application.** An exception normally disrupts the normal flow of the application; that is why we need to handle exceptions. Let's consider a scenario:

1. statement 1;
2. statement 2;
3. statement 3;
4. statement 4;
5. statement 5;//exception occurs
6. statement 6;
7. statement 7;
8. statement 8;
9. statement 9;
10. statement 10;

Suppose there are 10 statements in a Java program and an exception occurs at statement 5; the rest of the code will not be executed, i.e., statements 6 to 10 will not be executed. However, when we perform exception handling, the rest of the statements will be executed. That is why we use exception handling in [Java](https://www.javatpoint.com/java-tutorial).

### **Types of Java Exceptions**

There are mainly two types of exceptions: checked and unchecked. An error is considered as the unchecked exception. However, according to Oracle, there are three types of exceptions namely:

1. Checked Exception
2. Unchecked Exception
3. Error

## Q3 ->What are difference between Inheritance and Polymorphism?

## Inheritance: -

Inheritance is one in which a new class is created that inherits the properties of the already exist class. It supports the concept of code reusability and reduces the length of the code in object-oriented programming.  
Types of Inheritance are: 

1. Single inheritance
2. Multi-level inheritance
3. Multiple inheritance
4. Hybrid inheritance
5. Hierarchical inheritance

[**Polymorphism**](https://www.geeksforgeeks.org/polymorphism-in-c/)**:**   
Polymorphism is that in which we can perform a task in multiple forms or ways. It is applied to the functions or methods. Polymorphism allows the object to decide which form of the function to implement at compile-time as well as run-time.  
Types of Polymorphism are: 

1. Compile-time polymorphism (Method overloading)
2. Run-time polymorphism (Method Overriding)

| **S.NO** | **Inheritance** | **Polymorphism** |
| --- | --- | --- |
| 1. | Inheritance is one in which a new class is created (derived class) that inherits the features from the already existing class(Base class). | Whereas polymorphism is that which can be defined in multiple forms. |
| 2. | It is basically applied to classes. | Whereas it is basically applied to functions or methods. |
| 3. | Inheritance supports the concept of reusability and reduces code length in object-oriented programming. | Polymorphism allows the object to decide which form of the function to implement at compile-time (overloading) as well as run-time (overriding). |
| 4. | Inheritance can be single, hybrid, multiple, hierarchical and multilevel inheritance. | Whereas it can be compiled-time polymorphism (overload) as well as run-time polymorphism (overriding). |
| 5. | It is used in pattern designing. | While it is also used in pattern designing. |
| 6. | **Example:**  The class bike can be inherit from the class of two-wheel vehicles, which is turn could be a subclass of vehicles. | **Example:**  The class bike can have method name set\_color(), which changes the bike's color based on the name of color you have entered. |

### **Q4-> Explain various interfaces used in Collection framework?**

**Ans:-** Collection framework implements various interfaces, Collection interface and Map interface (java.util.Map) are the mainly used interfaces of Java Collection Framework. List of interfaces of Collection Framework is given below:

**1. Collection interface:** Collection (java.util.Collection) is the primary interface, and every collection must implement this interface.

**Syntax:**

1. public interface Collection<E>extends Iterable

Where <E> represents that this interface is of Generic type

**2. List interface:**List interface extends the Collection interface, and it is an ordered collection of objects. It contains duplicate elements. It also allows random access of elements.

**Syntax:**

**public** **interface** List<E> **extends** Collection<E>

**3. Set interface:** Set (java.util.Set) interface is a collection which cannot contain duplicate elements. It can only include inherited methods of Collection interface

**Syntax:**

**public** **interface** Set<E> **extends** Collection<E>

**Queue interface:**Queue (java.util.Queue) interface defines queue data structure, which stores the elements in the form FIFO (first in first out).

**Syntax:**

**public** **interface** Queue<E> **extends** Collection<E>

**4. Dequeue interface:** it is a double-ended-queue. It allows the insertion and removal of elements from both ends. It implants the properties of both Stack and queue so it can perform LIFO (Last in first out) stack and FIFO (first in first out) queue, operations.

**Syntax:**

**public** **interface** Dequeue<E> **extends** Queue<E>

**5. Map interface:**A Map (java.util.Map) represents a key, value pair storage of elements. Map interface does not implement the Collection interface. It can only contain a unique key but can have duplicate elements. There are two interfaces which implement Map in java that are Map interface and Sorted Map.

### **Q5->What is Thread in Java? What are the two ways of implementing thread in Java?**

**Ans->**Threads are basically the lightweight and smallest unit of processing that can be managed independently by a scheduler. Threads are referred to as parts of a process that simply let a program execute efficiently with other parts or threads of the process at the same time. Using threads, one can perform complicated tasks in the easiest way. It is considered the simplest way to take advantage of multiple CPUs available in a machine. They share the common address space and are independent of each other.

There are basically two ways of implementing thread in java as given below:

* Extending the **Thread** class

Example:

**class** **MultithreadingDemo** **extends** **Thread**

{

**public** **void** **run**()

{

System.out.println("My thread is in running state.");

}

**public** **static** **void** **main**(String args[])

{

MultithreadingDemo obj=**new** MultithreadingDemo();

obj.start();

}

}

Output:

My thread is in running state.

* Implementing **Runnable** interface in Java

Example:

**class** **MultithreadingDemo** **implements** **Runnable**

{

**public** **void** **run**()

{

System.out.println("My thread is in running state.");

}

**public** **static** **void** **main**(String args[])

{

MultithreadingDemo obj=**new** MultithreadingDemo();

Threadtobj =**new** Thread(obj); tobj.start();

}

}

Output:

My thread is in running state.