What is JAVA?

----------------------

Java is a high-level programming language and is platform-independent.

Java is a collection of objects. It was developed by Sun Microsystems. There are a lot of applications, websites, and games that are developed using Java.

What are the features of JAVA?

--------------------------------------

Features of Java are as follows:

OOP concepts

.Object-oriented

.Inheritance

.Encapsulation

.Polymorphism

.Abstraction

Platform independent: A single program works on different platforms without any modification.

High Performance: JIT (Just In Time compiler) enables high performance in Java. JIT converts the bytecode into machine language and then JVM starts the execution.

Multi-threaded: A flow of execution is known as a Thread. JVM creates a thread which is called the main thread. The user can create multiple threads by extending the thread class or by implementing the Runnable interface.

Why java is not 100%?

------------------------

Because of it's support primitive data types which are not objects -->boolean,byte,char,int,float,double,long,short

Wrapper classes

--------------------

A Wrapper classes is a class whose object wraps or contains primitive data types in other words we can wrap primitive values into wrapper classes they convert primitive data types into wraper classes

why pointers are not used in java?

--------------------------------------

.They are unsafe.

.Increases the complexity of the program and since java is known for its simplicity of code adding the concept of pointers will be contradicting

.since jvm is responsible for implicit memory allocation thus in order to avoid direct access to memory by the user

How does Java enable high performance?

----------------------------------------

Java uses Just In Time compiler to enable high performance. It is used to convert the instructions into bytecodes

What do you mean by Constructor?

-----------------------------------

Constructor can be explained in detail with enlisted points:

.When a new object is created in a program a constructor gets invoked corresponding to the class.

.The constructor is a method which has the same name as the class name.

.If a user doesn’t create a constructor implicitly a default constructor will be created.

.The constructor can be overloaded.

.If the user created a constructor with a parameter then he should create another constructor explicitly without a parameter.

What is meant by the Local variable and the Instance variable?

------------------------------------------------------------------

Local variables: are defined in the method and scope of the variables that exist inside the method itself.

Instance variable: is defined inside the class and outside the method and the scope of the variables exists throughout the class.

What is a Class?

---------------------

A class is a group of objects which have common properties. It is a template or blueprint from which objects are created. It is a logical entity. It can't be physical.

All Java codes are defined in a Class. It has variables and methods.

Variables are attributes which define the state of a class.

Methods are the place where the exact business logic has to be done. It contains a set of statements (or) instructions to satisfy the particular requirement

What is an Object?

--------------------------

An instance of a class is called an object. The object has state and behavior.

Whenever the JVM reads the “new()” keyword then it will create an instance of that class.

State: represents the data (value) of an object.

Behavior: represents the behavior (functionality) of an object such as deposit, withdraw, etc.

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

OOPS

------

Explain Is Java a pure Object Oriented language?

--------------------------------------------------------

Java is not an entirely pure object-oriented programming language. The following are the reasons:

.Java supports and uses primitive data types such as int, float, double, char, etc.

.Primitive data types are stored as variables or on the stack instead of the heap.

.In Java, static methods can access static variables without using an object, contrary to object-oriented concepts.

Why is a need for Object-oriented programming?

--------------------------------------------------

OOP provides access specifiers and data hiding features for more security and control data access, overloading can be achieved with function and operator overloading, Code Reuse is possible as already created objects in one program can be used in other programs.

Data redundancy, code maintenance, data security, and advantage of concepts such as encapsulation, abstraction, polymorphism, and inheritance in object-oriented programming provide an advantage over previously used procedural programming languages.

Explain Abstraction with a real-time example.

------------------------------------------------

Abstraction in object-oriented programming means hiding the implementation details and showing only functionality to the user.

In real life, an example of abstraction is an online shopping cart, say at any e-commerce site. Once you select a product and book order, you are just interested in receiving your product on time

example of ATM, the complexity of internals of how money is debited from your account is kept hidden, and you receive cash via a network. Similarly for cars, how petrol makes the engine run the automobile is extremely complex.

Give some real-time examples and explain Inheritance.

-----------------------------------------------------------------

Inheritance means one class (sub class) acquiring all the properties and behaviors of a parent object.(super class) by inheritance.

In real life, take an example of inheritance of a normal bicycle where it is a parent class and a sports bike can be a child class, where sports bike has inherited properties and behavior of rotating wheels with pedals via gears that of a normal bike.

Various types of inheritance are listed below:

Single Inheritance: Single child class inherits characteristics of the single-parent class.

Multiple Inheritance: One class inherits features of more than one base class and is not supported in Java, but the class can implement more than one interface.

Multilevel Inheritance: A class can inherit from a derived class making it a base class for a new class, for example, a Child inherits behavior from his father, and the father has inherited characteristics from his father.

Hierarchical Inheritance: One class is inherited by multiple subclasses.

Hybrid Inheritance: This is a combination of single and multiple inheritances.

How polymorphism works in Java, explain with real-life examples?

------------------------------------------------------------------------

A Polymorphism is an ability to have multiple forms or capability of the method to do different things. In real life, the same person performing different duties behaves differently. In-Office he is an employee, at home, he is a father, during or in after school tuitions he is a student, on weekends he plays cricket and is a player in the playground.

In Java, there are two types of polymorphism

Compile-time polymorphism: This is achieved by method overloading or operator overloading.

Runtime polymorphism: This is achieved by method overriding.

What is Interface?

-------------------

Interface is similar to the class where it can have methods and variables, but its methods do not have a body, just a signature known as the abstract method. Variables declared in the interface can have public, static, and final by default.

Interface is used in Java for achieve full abstraction and multiple inheritances, where the class can implement multiple interfaces.

What is Encapsulation

------------------------

Encapusulation is all neceessary data and methods are bind togeather and all the un-neccessary details are hidden to the normal user so encapusulation is process of binding data members and methods of a program togeather

Datahiding:Encapsulation is the process of hiding unwanted information such as restricting access to any member of an object

Databinding: encapsulation is process of binding the data members and the methods togeather as well as a class

Purpose of Encapsulation:

.Protects the code from others.

.Code maintainability.

Can you explain the advantages of Abstraction and Inheritance?

------------------------------------------------------------------

Abstraction reveals only essential details to the user and ignores or hides irrelevant or complex details. In other words, data abstraction exposes the interface and hides implementation details. Java performs abstraction with the help of interfaces and abstract classes. Advantage of abstraction is that it makes simple in viewing things by reducing or hiding the complexity of implementation.

Duplication of code is avoided, and it increases code reusability. Only essential details are revealed to the user and improves the security of the application.

Inheritance is where child class inherits functionality (behavior) of the parent class. We need not write code once written in parent class for functionality again in the child class and thus making it easier to reuse the code. The code becomes readable as well. Inheritance is used where there “is a” relation.

Example: Hyundai is a car OR MS Word is a software

What is the difference between extends and implements?

------------------------------------------------------------------------

Both extends and implements keyword are used for inheritance but in different ways.

Extends

---------------

A class can extend another class (child extending parent by inheriting his characteristics). Interface as well inherit (using keyword extends) another interface.

Sub class extending super class may not override all of the super class methods

Class can only extend a single super class.

Interface can extend more than one interfaces.

Syntax:class Child extends class Parent

Implements

----------------

A class can implement an interface

Class implementing interface has to implement all the methods of the interface.

Class can implement any number of interfaces.

Interface cannot implement any other interface.

Syntax:class Hybrid implements Rose

What are different access modifiers in Java?

-----------------------------------------------------

Access modifiers in Java controls access scope of class, constructor, variable, method, or data member. Various types of access modifiers are as follows:

.Default access modifier is without any access specifier data members, class and methods, and are accessible within the same package.

.Private access modifiers are marked with the keyword private, and are accessible only within class, and not even accessible by class from the same package.

.Protected access modifiers can be accessible within the same package or subclasses from different packages.

.Public access modifiers are accessible from everywhere.

Explain the difference between abstract class and method?

------------------------------------------------------------------------

Abstract Class

---------------------

Object cannot be created from the abstract class.

Sub class created or inherit abstract class to access members of abstract class.

Abstract class can contain abstract methods or non abstract methods.

Abstract Method

------------------

Abstract method has a signature but does not have a body.

It is compulsory to override abstract methods of super class in their sub class.

Class containing abstract method should be made abstract class.

What are the differences between method and constructor?

---------------------------------------------------------------------

Constructors

----------------

Constructors name should match with that of Class.

They are used to create, initialize and allocate memory to the object.

Constructors are implicitly invoked by the system whenever objects are created.

They are invoked using new keyword while creating an instance of the class (object).

Constructor does not have return type.

Constructor cannot be inherited by the subclass.

Methods

-------------

Methods should not have same name as Class name.

Methods are used to execute certain statements written inside them.

Methods are invoked when it is called.

Methods are invoked during program execution.

Method has a return type.

Methods can be inherited by a sub class.

When do you use the super keyword

----------------------------------------

The super keyword refers to superclass (parent) objects.

It is used to call superclass methods, and to access the superclass constructor.

The most common use of the super keyword is to eliminate the confusion between superclasses and subclasses that have methods with the same name.

.super can be used to refer immediate parent class instance variable.

.super can be used to invoke immediate parent class method.

.super() can be used to invoke immediate parent class constructor.

.Parent class method access can be done using super, when child class has method overridden.

When do you use this keyword

----------------------------------

The this keyword refers to the current object in a method or constructor. The most common use of the this keyword is to eliminate the confusion between class attributes and parameters with the same name

The this keyword can be used to refer current class instance variable. If there is ambiguity between the instance variables and parameters, this keyword resolves the problem of ambiguity

.Invoke current class constructor

.Invoke current class method

.Return the current class object

.Pass an argument in the method call

.Pass an argument in the constructor call

What is the difference between Runtime and compile-time polymorphism?

----------------------------------------------------------------------------------

Compile Time Polymorphism

-------------------------------

Call is resolved by a compiler in compile-time polymorphism.

It is also known as static binding and method overloading.

Same name methods with different parameters or methods with the same signature and different return types are compile-time polymorphism.

It is achieved by function and operator overloading.

As all the things are executed at compile time. compile-time polymorphism is less flexible.

Runtime Polymorphism

------------------------

Call is not resolved by the compiler in runtime polymorphism.

It is also known as dynamic, late, and method overriding.

Same name method with the same parameters or signature associated in different classes are called method overriding.

It can be achieved by pointers and virtual functions.

As things execute at run time, runtime polymorphism is more flexible.

What is method overloading?

-------------------------------

When two or more methods with the same name have either a different number of parameters or different types of parameters, these methods may have or may not have different return types, then they are overloaded methods, and the feature is method overloading. Method overloading is also called compile-time polymorphism.

What is method overriding?

-----------------------------

When a method of sub class (derived, child class) has the same name, parameters (signature), and same return type as the method in its super class (base, parent class) then the method in the subclass is said to be overridden the method in the superclass. This feature is also known as runtime polymorphism.

Explain constructor overloading.

----------------------------------------

More than one constructor with different parameters so that different tasks can be carried out with each constructor is known as constructor overloading. With constructor overloading, objects can be created in different ways.

Differentiate between static and dynamic binding?

---------------------------------------------------

Static Binding

----------------------

Static binding in Java use type of fields and class to as a resolution.

Method Overloading is an example of static binding.

Static binding gets resolved at compile time.

Methods and variables using static binding are private, final and static types.

Dynamic Binding

---------------------

Dynamic binding in Java uses object for resolving binding.

Method overriding is an example of dynamic binding.

Dynamic binding gets resolved at run time.

Virtual methods use dynamic binding.

What types of arguments can be used in Java?

----------------------------------------------------

For Java methods and functions, parameter data can be sent and received in different ways. If methodB() is called from methodA(), methodA() is a caller function and methodB() is called function, arguments sent by methodA() is actual arguments and parameters of methodB() is called formal arguments.

.Call By Value: Changes made to formal parameter (parameters of methodB()) do not get sent back to the caller (methodA()), This method is called call by value. Java supports the call by value.

.Call by Reference: Changes made to formal parameter (parameters of methodB()) are sent back to the caller (parameters of methodB()).

Any changes in formal parameters (parameters of methodB()) are reflected in actual parameters (arguments sent by methodA()). This is called call by reference.

Difference between String, String Builder, and String Buffer.

-----------------------------------------------------------------

String: String variables are stored in a “constant string pool”. Once the string reference changes the old value that exists in the “constant string pool”, it cannot be erased.

Example: String name = “book”;

String Buffer:

--------------------

Java StringBuffer class is used to create mutable (modifiable) String objects. The StringBuffer class in Java is the same as String class except it is mutable

Here string values are stored in a stack. If the values are changed then the new value replaces the older value.

The string buffer is synchronized which is thread-safe.

Performance is slower than the String Builder.

Example:String Buffer name =”book”;

Note:Once the name value has been changed to “pen” then the “book” is erased in the stack.

String Builder:

-------------------

This is the same as String Buffer except for the String Builder which is not threaded safely that is not synchronized. So obviously the performance is fast.

Why String is immutable in java?

------------------------------------

.String pool requires string to be immutable otherwise shared reference can be changed from anywhere

.Secure because string is shared on different area like file system,networking connection,database connection,having immutable string allows you to be secure and safe because no one can change reference of string once it gets created

.you con't modified once string gets created.

What is a marker interface?

-------------------------------

.A marker interface can be defined as the interface having no data members and member dunction.in simpler terms an empty interface is called the marker interface

EX: Serilizable,clonable

public interface clonable {

}

Can you ovveride a private or static methods?

---------------------------------------------------

.you can not ovveride a privete or static method

.you cannot ovveride a private method in sub class because it's not accessible there what you do is create another private method with the same name in child class

.for static methods if you create a similar method with same return type and same method arguments in child class then it will hide the super class method this is known as method hiding

Does "finnaly" always execute in java?

-------------------------------------------------

Not in following cases

.System.exit() function

.System crash

. even try catch happen or not finally block will execute.

How can you make a class immutable?

---------------------------------------

.declare the class as final so its can't be executed.

.make all fileds private so that direct access is not allowed

.don't provide setter methods for variable

.make all mutable fileds final so that it's value can be assigned only once

.Initialize all the fields via a constructor performing deep copy.

What is singleton class in java and howcan we make a class singleton?

-------------------------------------------------------------------------

.A singleton class in java can have only one instance can be created at any given time in one jvm and hence all it's methods and variable belong to just one instance

.making the constructor as private

.creating the static instance with the type same as that of class.

.creating a static method i.e. getInstance(), in that return the same instance if it is not null or else create a new instance.

.thisis useful for the situations when there is a need to limit the no of objects for a class

Example: singleton usage scenario is when there is a limit of having only one connection to a database due to some driver limitations or licensing issues

Ex:

public class Animal{

private static Animal single\_instance = null;

private Animal(){

}

public static Animal getInstance(){

if(single\_instance == null)

single\_instance = new Animal();

return single\_instance;

}

}

Difference between HAS-A and IS-A

---------------------------------------

a Has-A relationship is also known as composition. ... In Java, a Has-A relationship simply means that an instance of one class has a reference to an instance of another class or an other instance of the same class. For example, a car has an engine, a dog has a tail and so on.

In Is-A relationship one class is obtaining the features of another class by using inheritance concept with extends keywords.

Difference between abstract class and interface

---------------------------------------------------

Abstract class

----------------------

Abstract class can have abstract and non-abstract methods.

Abstract class doesn't support multiple inheritance.

Abstract class can have final, non-final, static and non-static variables.

Abstract class can provide the implementation of interface.

The abstract keyword is used to declare abstract class.

An abstract class can extend another Java class and implement multiple Java interfaces.

An abstract class can be extended using keyword "extends".

A Java abstract class can have class members like private, protected, etc.

Example:

public abstract class Shape{

public abstract void draw();

}

Interface

---------------

Interface can have only abstract methods. Since Java 8, it can have default and static methods also.

Interface supports multiple inheritance.

Interface has only static and final variables.

Interface can't provide the implementation of abstract class.

The interface keyword is used to declare interface.

An interface can extend another Java interface only.

An interface can be implemented using keyword "implements".

Members of a Java interface are public by default.

Example:

public interface Drawable{

void draw();

}

Association

--------------

Association refers to the relationship b/w multiple objects it refers to how objects are related to each other and how they are using each other's functionallity

Types

.Composition

.Aggregation

Composition

-------------

The composition is the strong type of association an association is said to composition if an object owns another object and another object cnnot exist without owner object

Aggregation

-----------

Aggregation is a weak association an association is said to be aggregation if both objects can exist independently

Difference between Aggregation and Composition

---------------------------------------------------

Aggregation

------------

Aggregation is a part of an association relationship.

Aggregation is considered as a weak type of association.

In an aggregation relationship, objects that are associated with each other can remain in the scope of a system without each other.

In Aggregation, linked objects are not dependent upon the other object.

Composition

--------------

The composition is a part of an association relationship.

The composition is considered as a strong type of association.

In a composition relationship, objects that are associated with each other cannot remain in the scope without each other.

In composition, objects are highly dependent upon each other.

Final Keyword In Java

-------------------------

The final keyword in java is used to restrict the user. The java final keyword can be used in many context.

Java final variable

------------------------

If you make any variable as final, you cannot change the value of final variable

Java final method

------------------

If you make any method as final, you cannot override it.

Java final class

----------------------

If you make any class as final, you cannot extend it.

Java static keyword

------------------------

The static keyword in Java is used for memory management mainly

Java static variable

--------------------

The static variable can be used to refer to the common property of all objects

.Memory efficient

Java static Method

--------------------

.static method belongs to the class rather than the objects

.There is no need to create the objects to call the static methods

.A static method can access and change the value of the static variable

.when there is requirement to share a method or a variable b/w multiple objects of a class instead of creating separate copies of each object we use static keyword to make a method or varible shared for all objects

Collections

===========================

What is the meaning of Collections in Java?

----------------------------------------------

Collection is a framework that is designed to store the objects and manipulate the design to store the objects.

Collections are used to perform the following operations:

.Searching

.Sorting

.Manipulation

.Insertion

.Deletion

A group of objects is known as collections. All the classes and interfaces for collecting are available in Java util package.

What are all the Classes and Interfaces that are available in the collections?

----------------------------------------------------------------------------------

Given below are the Classes and Interfaces that are available in Collections:

Interfaces:

.Collection

.List

.Set

.Map

.Sorted Set

.Sorted Map

.Queue

Classes:

.Lists:

.Array List

.Vector

.Linked List

Sets:

.Hash set

.Linked Hash Set

.Tree Set

Maps:

.Hash Map

.Hash Table

.TreeMap

.Linked Hashed Map

Queue:

.Priority Queue

Difference between Array and Array List.

----------------------------------------------------------

Array

-----------------

.Size should be given at the time of array declaration.

.String[] name = new String[2]

.To put an object into array we need to specify the index.

name[1] = “book”

.Array is not type parameterized

Array List

------------------

.Size may not be required. It changes the size dynamically.

.ArrayList name = new ArrayList

.No index required.

name.add(“book”)

.ArrayList in java 5.0 are parameterized.

Eg: This angle bracket is a type parameter which means a list of String.

Explain the different lists available in the collection.

------------------------------------------------------------

Values added to the list are based on the index position and it is ordered by index position. Duplicates are allowed.

A)Array List:

---------------

.Fast iteration and fast Random Access.

.It is an ordered collection (by index) and not sorted.

.It implements the Random Access Interface.

b) Vector:

---------------

It is the same as Array List.

Vector methods are synchronized.

Thread safety.

It also implements Random Access.

Thread safety usually causes a performance hit.

c) Linked List:

------------------

Elements are doubly linked to one another.

Performance is slower than the Array list.

Good choice for insertion and deletion.

In Java 5.0 it supports common queue methods peek( ), Pool ( ), Offer ( ) etc.

Explain about Set and their types in a collection.

--------------------------------------------------------

Set cares about uniqueness. It doesn’t allow duplications. Here “equals ( )” method is used to determine whether two objects are identical or not.

a) Hash Set:

-----------------

Unordered and unsorted.

Uses the hash code of the object to insert the values.

Use this when the requirement is “no duplicates and don’t care about the order”.

b) Linked Hash set:

---------------------

An ordered version of the hash set is known as Linked Hash Set.

Maintains a doubly-Linked list of all the elements.

Use this when an iteration order is required.

c) Tree Set:

-----------------

It is one of the two sorted collections.

Uses the “Read-Black” tree structure and guarantees that the elements will be in ascending order.

We can construct a tree set with the constructor by using a comparable (or) comparator.

Explain about Map and its types.

---------------------------------------

Answer: Map cares about the unique identifier. We can map a unique key to a specific value. It is a key/value pair. We can search a value, based on the key. Like the set, the map also uses the “equals ( )” method to determine whether two keys are the same or different.

Map is of following types:

a) Hash Map:

------------------

Unordered and unsorted map.

Hashmap is a good choice when we don’t care about the order.

It allows one null key and multiple null values.

b) Hash Table:

-------------------

Like the vector key, methods of the class are synchronized.

Thread safety and therefore slows the performance.

It doesn’t allow anything that is null.

c) Linked Hash Map:

-----------------------

Maintains insertion order.

Slower than Hash map.

I can expect a faster iteration.

d) TreeMap:

-------------

Sorted Map.

Like Tree set, we can construct a sort order with the constructor.

Difference between HashMap and HashTable.

--------------------------------------------------

HashMap

-------------

Methods are not synchronized

Not thread safety

Iterator is used to iterate the values

Allows one null key and multiple null values

Performance is high than HashTable

HashTable

--------------

Key methods are synchronized

Thread safety

Enumerator is used to iterate the values

Doesn’t allow anything that is null

Performance is slow

Difference between HashSet and TreeSet.

-------------------------------------------------

HashSet

Inserted elements are in random order

Can able to store null objects

Performance is fast

TreeSet

-------------

Maintains the elements in the sorted order

Couldn’t store null objects

Performance is slow

What is concurrent Hash Map?

----------------------------

.concurrent hashmap class is thread-safe multiple thread can operate on single object without any complication

.At a time any no of threads are appicable for a read operation without locking the concurrent hash map

.default conncurrent-level of concurrent hash map is 16

Difference between fail-fast and fail-safe iterators?

------------------------------------------------------------

fail-fast iterator will throw an Concurrent modification exception immediately whenever they encounter that one thread is modifying the Collection

object while other thread is iterating over that object.

fail-safe interator doesnt throw any exception even if collection is modified structurally when other thread is iterating over it because they work

on clone of Collection instead of working on original Collection.

Difference between synchronized Collection and Concurrent Collection?

=====================================================================

In Synchronized Collection, if one thread is performing one segment in the entire collection, all other threads will remain calm

until the thread1 will completed its task.

Whereas in concurrent collection, threads will perform their tasks concurrently without having to wait for other threads to complete their own tasks.

Intenal working of hash map?

---------------------------------

.Hash map in java works on hashing principle where hash functions are used to link key and values in hash map objects(map.entry-->contains key,value) are stored by calling put(key,value) method of hashmap and retrived by calling get(key)

.when we call put method hashcode() method of the key object is called which calculates an index of the bucket location where we can store the value object

. to retrive you call theget() method and again pass the key object which lands you up at same index or bucket and you retrive the value object.

or

hashing is the technique where the object is converted into integer value. this integer value is used for indexing and faster searches.

It internally uses 2 methods,

equals()- method of Object class. it compares the equality of 2 objects. it compares the key whether they are equal or not.

hashcode()-method of Object class.

bucket is normally an array of nodes. nodes will act as an array and linkedlist datastructure.

it will calcuates the index value into which the object has to be inserted based on the hashcode of the objects.

Hashing collision: if two objects has got the same index into which it has to be inserted, then equals method will compare the keys of

both, if matches then it replace the already present one, if not it will be stored along with the older one with the help of linkedlist.

What is Hash map?

--------------------

.Hashmap is a part of the java collection framework it uses a technique called hashing it implements the map interface it stores the data in the pairof key and value

.hash map contains array of nodes and the node is represened as a class

.it uses an array and linkedlist data structure internally for storing key and value

Exception Handling

==========================

What is meant by Exception?

----------------------------------

An Exception is a problem that can occur during the normal flow of execution. A method can throw an exception when something wails at runtime. If that exception couldn’t be handled, then the execution gets terminated before it completes the task.

What is error

--------------------

.An error is an ir-recoverable condition occuring at runtime such as outofmemory error

.these JVM errors you cannot repair them at runtime

What are the types of Exceptions?

----------------------------------------------

a) Checked Exception:

-------------------------------

These exceptions are checked by the compiler at the time of compilation. Classes that extend Throwable class except Runtime exception and Error are called checked Exception.

Checked Exceptions must either declare the exception using throws keyword (or) surrounded by appropriate try/catch.

For Example, ClassNotFound Exception

b) Unchecked Exception:

--------------------------

These exceptions are not checked during the compile time by the compiler. The compiler doesn’t force to handle these exceptions. It includes:

Arithmetic Exception

ArrayIndexOutOfBounds Exception

What are the different ways to handle exceptions?

---------------------------------------------------------

a) Using try/catch:

----------------------

The risky code is surrounded by try block. If an exception occurs, then it is caught by the catch block which is followed by the try block.

b) By declaring throws keyword:

------------------------------------

At the end of the method, we can declare the exception using throws keyword.

EX: public void add() throws Exception{

What are the advantages of Exception handling?

---------------------------------------------------------

The normal flow of the execution won’t be terminated if an exception gets handled

We can identify the problem by using catch declaration

Threads

===========

What is a Thread?

----------------------

In Java, the flow of execution is called Thread. Every java program has at least one thread called the main thread, the main thread is created by JVM. The user can define their own threads by extending the Thread class (or) by implementing the Runnable interface. Threads are executed concurrently.

How do you make a thread in Java?

-------------------------------------

a) Extend Thread class: Extending a Thread class and override the run method. The thread is available in java.lang.thread.

b) Implement Runnable interface: Another way is by implementing the runnable interface. For that, we should provide the implementation for the run () method which is defined in the interface.

Explain about join () method.

------------------------------------

Join () method is used to join one thread with the end of the currently running thread.

yield ()

------------

A yield () method moves the currently running thread to a runnable state and allows the other threads for execution. So that equal priority threads have a chance to run

Explain about wait () method.

-----------------------------------

wait () method is used to make the thread to wait in the waiting pool. When the wait () method is executed during a thread execution then immediately the thread gives up the lock on the object and goes to the waiting pool. Wait () method tells the thread to wait for a given amount of time.

Then the thread will wake up after notify () (or) notify all () method is called.

Difference between notify() method and notifyAll() method in Java.

------------------------------------------------------------------------------

notify()

This method is used to send a signal to wake up a single thread in the waiting pool.

notifyAll()

This method sends the signal to wake up all the threads in a waiting spool.

How to stop a thread in java? Explain about sleep () method in a thread?

-------------------------------------------------------------------------------

We can stop a thread by using the following thread methods:

Sleeping

Waiting

Blocked

Sleep: Sleep () method is used to sleep the currently executing thread for the given amount of time. Once the thread is wake up it can move to the runnable state. So sleep () method is used to delay the execution for some period.

It is a static method.

When to use the Runnable interface Vs Thread class in Java?

-------------------------------------------------------------------

If we need our class to extend some other classes other than the thread then we can go with the runnable interface because in java we can extend only one class.

If we are not going to extend any class then we can extend the thread class.

Difference between start() and run() method of thread class.

------------------------------------------------------------------

Start() method creates a new thread and the code inside the run () method is executed in the new thread. If we directly called the run() method then a new thread is not created and the currently executing thread will continue to execute the run() method.

t.start()-->a new thread will be created which is responsible for the execution of run() method

t.run() -->no new hread will be created and run() method will be executed just like a normal method by the main method.

What is Multi-threading?

-------------------------------

Multiple threads are executed simultaneously. Each thread starts its own stack based on the flow (or) priority of the threads.

To run multiple tasks in a concurrent manner within single program threads share same process stack and running in parallel

it helps in performane improvement of any program

Life cycle of Thread?

-----------------------

New: When you create a thread object and it is not alive yet

Runnable: When you call start() of thread it goes into runnable state wheather it will execute immediately or execute after some time depends on thread scheduler

Running: When thread is being execute it goes to running state

Blocked or Non-Runnable: When thread waits for some other resources or some other thread to complete it goes to blocked state (due to thread's join)

Dead: Once the run method is completed then it is terminated. Now the thread is not alive.

What is Synchronization?

---------------------------------

Synchronization makes only one thread to access a block of code at a time. If multiple threads accesses the block of code, then there is a chance for inaccurate results at the end. To avoid this issue, we can provide synchronization for the sensitive block of codes.

The synchronized keyword means that a thread needs a key in order to access the synchronized code.

What is meant by Serialization?

-----------------------------------

Converting a file into a byte stream is known as Serialization. The objects in the file are converted to bytes for security purposes. For this, we need to implement a java.io.Serializable interface. It has no method to define.

Variables that are marked as transient will not be a part of the serialization. So we can skip the serialization for the variables in the file by using a transient keyword.

What is the purpose of a Volatile Variable?

-----------------------------------------------

Volatile variable values are always read from the main memory and not from thread’s cache memory. This is used mainly during synchronization. It is applicable only for variables.

Difference between Serialization and Deserialization in Java.

---------------------------------------------------------------------

Serialization

-----------------

Serialization is the process which is used to convert the objects into byte stream

An object is serialized by writing it an ObjectOutputStream.

Deserialization

----------------------

Deserialization is the opposite process of serialization where we can get the objects back from the byte stream.

An object is deserialized by reading it from an ObjectInputStream.

https://www.softwaretestinghelp.com/core-java-interview-questions/

https://www.softwaretestinghelp.com/oops-interview-questions-and-answers/

https://www.guru99.com/java-interview-questions-answers.html

https://www.interviewbit.com/java-interview-questions/