JAVA QUESTIONNAIRE

1. What is constructor?

- constructor is a block of codes, similar to the method
- constructor name must same as the class name.
- When an object for the class is created, then the execution of the constructor takes place automatically.
- The execution of constructor always provides an object/instance.

There are three types of constructors

a) Default constructor:> Every class in a core java by default it contains default constructor, it can be used for accessing the instance members.
Here, the constructor will be not defined explicitly.

b) No-args constructor:> This type of constructor as explicitly specified in a class. This type of constructor never accepts parameter/arguments.
In this constructor we are hardcoding the values. We usually prefer this constructor when we require the same kind of Output.

c)Parametrized constructor:> This type of constructor as explicitly specified in a class. This type of constructor accepts parameter.
In this constructor we are remove the hardcoding of value. We usually prefer this constructor when we require the different outputs.

2**. Explain Constructor Overloading?**

--> The process of defining multiple constructors of the same class is referred to as Constructor overloading.
The constructors overloading can be achieved by specifying multiple constructor in a same class, each parameter must differ with number of parameters, datatype of each parameter, sequence of datatype of each parameter.

3. Explain Encapsulation concept?

- →Encapsulation is one of the oops concept in core java, which indicates data hiding.
- We can implement encapsulation using access specifiers (private)
- Encapsulation is achieved by getter/setter approach.
- Encapsulation solves problems at implementation level

4. Difference between call by value and Call by Reference?

No.	Call By Value	Call by Reference
1)	There is a involvement of primitive datatype	There is a involvement of derived datatype
2)	Any changes which take place within the body, those changes never reflect outside	Any changes which take place within the body, those changes reflect outside
3)	Call By value means calling a method with a parameter as a value.	Call By reference means calling a method with a parameter as a reference

5. Difference between final and Finally keywords?

No.	Final Keyword	Finally Keyword
1)	Final is the Keyword which is used to apply restrictions on a class, method or variable	Finally is optional block, irrespective of execution has arised or not, this block executes always.
2)	Since declared final, variable becomes constant and cannot be modified	Since their block executes by default so we can provides the statement closing, the files disconnecting rom the DB…etc.
3)	Final method cannot be overridden by subclass	Finally block is executed as soon as try-catch block executes.

6. Difference between throw and throws in Exception handling?

Throw:->it is used to throw an exception explicitly in the code, inside the function or the block of code.

- The throw is used within the method.
- We are allowed to throw only one exception at a time.(ex: throw new io -exception)

- The throw keyword is followed by an instance of exception to be thrown.

  Throws:-> it is used in method signature to declare an exception which might be thrown by the function while the execution of the code.
- Throw's is used with the method signature.
- We can declare multiple exceptions using throws(ex: main() throws IO exception, sql exception.
- The throws keyword is followed by class name of exception to be thrown.

7. Difference between String Buffer and String Builder?

No.	String Buffer	String Builder
1)	String Buffer is a mutable(alter), it means it accepts the modification	String builder is also a mutable(alter), it means it accepts the modification
	String Buffer is *synchronized* i.e. **thread safe**. It means two threads can't call the methods of String Buffer simultaneously.	String Builder is *non-synchronized* i.e. **not thread safe**. It means two threads can call the methods of String Builder simultaneously.
2)	String Buffer is *less efficient* than String Builder.	String Builder is *more efficient* than String Buffer.
3)	String Buffer was introduced in Java 1.0	String Builder was introduced in Java 1.5

8. Difference between Checked Exception and Unchecked Exception?

No.	Checked Exception	Unchecked Exception
1)	Checked exception occur at compile time	Unchecked Exception occur at runtime
2)	The Compiler checks the checked exceptions	Compiler does not check the unchecked exception.

3)	These types of exceptions can be handled at the time of compilation	These types of exceptions cannot be a catch/handle at the time of compilation, because they get generated by the mistake in the program
4)	Here, the jvm needs the exception to catch & handle	Here, the jvm does not require the exception to catch & handle
5)	Examples: FileNotFoundException NoSuchFieldException InteruptedException NoSuchMethodException ClassNotFoundException IoException	Examples: ArithemeticException indexOutOfBoundException NullPointException NoSuchElementException NumberFormetException SecurityException EmptyStackEception

9. Difference between List and Set interfaces?

No.	List Interface	Set Interface
1)	The List is an indexed sequence.	The Set is an non-indexed sequence.
2)	List allows duplicate elements	Set doesn't allow duplicate elements.
3)	Elements by their position can be accessed.	Position access to elements is not allowed.
4)	Multiple null elements can be stored.	Null elements can be store only once.
5)	List Implementations are Array List, Linked List, Vector List.	Set Implementations are HalsSet,LinkedHasSet.

11. Explain Outer class and Inner class concept?

In java the classes can be nested (outer class and inner class) in this case, the members of the outer class can be directly accessed by the inner class.

But members of the inner class cannot be access outer class directly, In order to access in outer class we have to create a object or instance for the inner class.

12. Explain possible approaches of Execution of Static Block?

Execution of a static block can be achieved by following three approaches:

a) Create an object for the respective class containing static block.
b) The static block executes prior execution of any other members.
c) If the static block is available in an independent class, we can execute it by calling
Class.methodName();

13. Explain Abstract class and its specific features?

Abstract class provides partial implementation ,if the multiple subclass needs to implement same behaviour then we need to prefer abstract class.

Features:

- Abstract class should contains an abstract method.
- If a class contains abstract method, then the class must be specified as an abstract.
- If a class does not contains abstract method even though we can specify the class as an abstract.
- The abstract methods contains only the signature and without body.
- In abstract class for abstract methods the abstract keyword is mandatory.
- At any point the static methods we can't able to define as an abstract or abstract methods cannot be a static.
- If a subclass is extending the abstract class then the subclass responsibility it has to implements all the abstract methods from the super class . If a subclass has missed to implement any abstract method from the super class the subclass also becomes an abstract class.
- We can't create an object or instance for the abstract class alone, we can provide only the object reference. In this case, only the instance Members which are available in the abstract class alone can be accessible.
- An abstract class contains abstract methods, instance members and static members.

14. Explain Interface and its specific features?(wrf 1.7 Version)

The interface gives the complete implementation in order to implement different behaviour from different component we have to prefer the interface.

Its Features:

- Interface contains only the abstract methods.
- In interface abstract keyword is optional for abstract methods.
- The subclass always use “implements” keyword for implementing the interface.
- When a subclass implements abstract method from interface in subclass it must be public access specifier for those methods.
- A Single subclass can implement more than one interface that indicates multiple inheritance.
- An interface can extends one more interface at any level.
- If a subclass implementing an interface the subclass responsibility it has to implement all the abstract methods from the interface. If subclass missed to implements any abstract methods from the interface the subclass becomes an abstract class.
- We can’t create a object for a interface, we can give only for object reference ,in this case only the members which are available in the interface alone can be accessible.
- If we would like to declare variables in interface those variables by default becomes final and static.

Why we need interface:

* It is used to achieve total abstraction.
* Since java does not support multiple inheritances in the case of class, by using an interface it can achieve multiple inheritances.
* It is also used to achieve loose coupling.
* Interfaces are used to implement abstraction.

15. Difference between Byte Stream and Character Stream?

Character Stream	Byte Stream
A character stream will access a file character by character.	A byte stream access the file byte by byte.
These handle data in 16 bit Unicode. The character stream classes read/write 16-bit characters.	These handle data in bytes (8 bits) i.e., the byte stream classes read/write data of 8 bits.
Using these you can read and write text data only.	Using these you can store characters, videos, audios, images etc.
A character stream needs to be given the file's encoding in order to work properly.	byte stream do not use any encoding.

16. Explain the Polymorphism concept?

A Super class reference variable can refers each child class object based on the super class reference variable we can execute the subclass method. That is called as dynamic method dispatch or polymorphism.

17. Explain Singleton design Pattern?

The Single ton is a design pattern, the single ton is allows creates only one object or instance, if you try to create one more object instead of creating new one it returns the existing object.

Steps to apply single ton:
1.Define the no agrs constructor as a private members of the class can be accessible within the same class.
2.Write a static method, the static method has to must return the object.
3.Write a logic in a static method, if reference variable is null, then only it has to allocates the memory, otherwise it has to returns the existing object.

18. Describe the Debug process in Eclipse?

The debugging can be achieved by applying toggle breakpoint, based on applying the toggle break, The eclipse can enter into the debug mode.

Once after enter into the debug mode, we can perform the below actions.

- Step Into - F5
  Executes the current line of code and dives into the next line of code in

the program execution. If the current line calls a method, the debugger steps into the method.

- Step Over- F6
  Executes the current line of code and goes to the next line without stepping into any method calls or associated scope (e.g. loops and conditions) of the current line.
- Step Return-F7
  Steps back out of the current method and returns to the caller of the method

19. Explain Method Override concept?

Writing same method in both parent and child class in which method name, signature and return type are same in both parent and child class.

Here child class method will hide the super class method which means when you create a object to the child class only child class method will execute by hiding or overriding super class method
It can achieved by using “super” keyword.

20. Explain Method Overloading?

Method Overloading can be achieved by using same method name for multiple times, each method differ with

1. The number of parameters.
2. The data types of the parameters.
3. The sequence of datatype of the parameters.

based on these approaches we can overload the method.

21. **Explain marker interface?**

22. List out the interfaces you have used in Core Java?

- Collection
- List
- Set
- Queue
- Iterable
- Sorted Set

23. List out the abstract class you have used in Core Java?

- Input Stream
- Output Stream
- Writer
- Reader
- Abstract collection
- Abstract Set
- Abstract List

24. Explain the different types of Inheritance in Core Java?

Inheritance is an important pillar of OOPs (Object-Oriented Programming).

Inheritance provides code reusability, It is the mechanism in java by which one class is allowed to inherit the features (fields and methods) of another class. (**by child class object we can access all the members of superclass**)

Inheritance can be achieved by using "extends" keyword.

Types of Inheritance in Java

Below are the different types of inheritance which are supported by Java.

1. Single Inheritance: In single inheritance, subclasses inherit the features of one superclass. In the image below, class A serves as a base class for the derived class B.

2. Multilevel Inheritance: In Multilevel Inheritance, a derived class will be inheriting a base class and as well as the derived class also act as the base class to other class. In the below image, class A serves as a base class for the derived class B, which in turn serves as a base class for the derived class C. In Java, a class cannot directly access the grandparent's members.

3. Hierarchical Inheritance: In Hierarchical Inheritance, one class serves as a superclass (base class) for more than one subclass. In the below image, class A serves as a base class for the derived class B, C and D.

4. Hybrid Inheritance(Through Interfaces): It is a mix of two or more of the above types of inheritance. Since java doesn't support multiple inheritances with classes, hybrid inheritance is also not possible with classes. In java, we can achieve hybrid inheritance only through Interfaces.

5. Multiple Inheritance (Through Interfaces): In Multiple inheritances, one class can have more than one superclass and inherit features from all parent classes. Please note that Java does **not** support multiple inheritances with classes. In java, we can achieve multiple inheritances only through Interfaces. In the image below, Class C is derived from interface A and B.

26. Explain multithreading concept in core java?

It is the ability of a machine to execute multiple threads independently at the same time but share the process resources simultaneously. Its main purpose is to provide simultaneous execution of multiple threads to utilize the machine time as much as possible.

Here we create multiple threads, for each thread we are assigning the same job.

//Various benefits of Multithreading//

- Allow the program to run continuously even if a part of it is blocked.
- Improve performance as compared to traditional parallel programs that use multiple processes.
- Allows to write effective programs that utilize maximum CPU time
- Improves the responsiveness of complex applications or programs.
- Increase use of CPU resources and reduce costs of maintenance.
- Saves time and parallelism tasks.
- If an exception occurs in a single thread, it will not affect other threads as threads are independent.
- Less resource-intensive than executing multiple processes at the same time.

27.Difference between Queue and Stack in Java.Util?

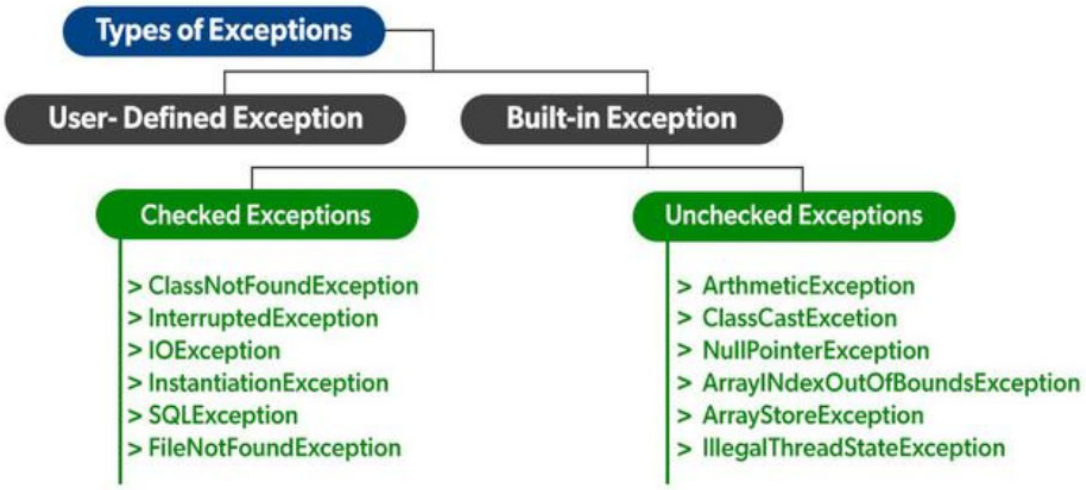
Stacks	Queues
Stacks are based on the LIFO principle, i.e., the element inserted at the last, is the first element to come out of the list.	Queues are based on the FIFO principle, i.e., the element inserted at the first, is the first element to come out of the list.
Insertion and deletion in stacks takes place only from one end of the list called the top.	Insertion and deletion in queues takes place from the opposite ends of the list. The insertion takes place at the rear of the list and the deletion takes place from the front of the list.
Insert operation is called push operation.	Insert operation is called enqueue operation.
Delete operation is called pop operation.	Delete operation is called dequeue operation.
In stacks we maintain only one pointer to access the list, called the top, which always points to the last element present in the list.	In queues we maintain two pointers to access the list. The front pointer always points to the first element inserted in the list and is still present, and the rear pointer always points to the last inserted element.
Stack is used in solving problems works on recursion.	Queue is used in solving problems having sequential processing.

28.How to return an object or an instance from a method?

```
class Test
   {
        public static Test obj=null;
        private Test()
        {
        }
        public static Test getinstance()
        {
            if (obj==null)
            {
                obj=new Test;
            }
        return obj;
        }
   }
```

29. How to do Casting? Explain some casting Scenarios you have done in core Java?

30. List out some common Exception classes you have experienced in Core Java?



31.Differnce between Abstract Class and Interface in Core Java?

- ***Type of methods*:** Interface can have only abstract methods. An abstract class can have abstract and non-abstract methods.
- ***Final Variables:*** Variables declared in a Java interface are by default final. An abstract class may contain non-final variables.
- ***Type of variables:*** Abstract class can have final, non-final, static and non-static variables. The interface has only static and final variables.
- ***Implementation:*** Abstract class can provide the implementation of the interface. Interface can't provide the implementation of an abstract class.
- ***Inheritance vs Abstraction:*** A Java interface can be implemented using the keyword "implements" and an abstract class can be extended using the keyword "extends".
- ***Multiple implementations:*** An interface can extend another Java interface only, an abstract class can extend another Java class and implement multiple Java interfaces.
- ***Accessibility of Data Members:*** Members of a Java interface are public by default. A Java abstract class can have class members like private, protected, etc.

32. Describe Constant Pool memory for String Literals in Core Java?

Constant Pool is memory section which is available in HEAP MEMORY, the constant pool never accepts duplicate objects, before creation of new object 1st it verifies the availability of the object. If the object exists instead of creating new object it provides the same reference.

33. Describe Serialization & Deserialization in Core Java?

Serialization is a mechanism of converting the state of an object into a byte stream. Here the file must have extension .scr.

OR

Serialization is the process of converting an object from java supported form into a file-supported form or network-supported form by fileoutputStream and objectoutputStream classes we can implement serialization.

Deserialization is the reverse process where the byte stream is used to recreate the actual Java object in memory. This mechanism is used to persist the object.