1. WHAT IS INTERFACE?

* It is used to define the significance of a method and it’s a blue print of a class.
* It contains only abstract methods
* It won’t contain any method body and any instance variables.
* In this, methods will be available but method implementation will not be there.
* We cannot create the object of an interface because of no implementation methods present in interface.
* Access specifiers for methods in interface must be public
* Variables defined must be public, static and final
* Implements keyword is used

1. WHAT IS ABSTRACT CLASS/METHOD?

* It contains both abstract methods and concrete methods (Implemented).
* If there is at least one abstract method in a class, that will be an abstract class.
* It should be defined with name of “abstract” keyword.
* We cannot make abstract class as final.
* We cannot create object for abstract class and can create object in corresponding sub class and can call the methods.
* Extends keyword is used
* Except private we can have any access specifier for methods in abstract class
* Except private variables we can have any access specifiers

1. HOW TO ACHIEVE ABSTRACTION?

* Abstraction is process of hiding the implementation details and showing only functionality to the user.

Two ways to achieve abstraction:

* Implement the method and only to allow to access the method.
* Don’t implement the method, Ask to subclass or implementing class to implement the method according to their convenient.

1. WHAT IS ENCAPSULATION?

* Encapsulation in Java is a mechanism of wrapping the data (variables) and code acting on the data (methods) together as a single unit of class.
* In encapsulation, the variables of a class will be hidden from other classes, and can be accessed only through the methods of their current class. Therefore, it is also known as data hiding.

Achieve encapsulation in Java.

* Declare the variables of a class as private.
* Provide public setter and getter methods to modify and view the variables values.

1. WHAT IS POLYMORPHISM AND TYPES?

* Performing single object in multiple ways, simply achieving method over loading and overriding concepts.

1. WHAT IS METHOD OVERLOADING?

Method signature= methodname(Parameters) .

* If we have multiple methods in a class with same name with different parameters and datatypes this process is called as method overloading.

1. WHAT IS METHOD OVERRIDING?

* If we want to declare and implement a method in a sub class that method is already available in the super class. This process is called as method overriding.

1. WHAT IS INHERITANCE?

* Inheritance is process of acquiring the properties from one class to another class or extending all methods and variables from super class to sub class, using EXTENDS keyword.
* Multiple inheritance is not allowed in Java. Multiple inheritance can be achieved using interface

Reason:

1. WHAT IS CONSTRUCTOR AND TYPES OF CONSTRUCTOR?

* Constructor is a special type of method; it is invoked at the time of object creation.
* In this class name and method name will be same. No return type for constructors
* We can overload the constructor but we cannot override the constructor.
* We cannot make constructor as final.
* We have two types of constructors.

Parameterized constructor:

* If we pass any parameters inside of the constructor, that will be the parameterized constructor.

Default constructor:

* If we don’t pass any parameters inside of the constructor, that will be default constructor.
* It prints 0, null.

1. WHAT IS THE PURPOSE OF DEFAULT AND PARAMETERISED CONSTRUCTOR?

* While initiating the object, if you want write /do some logic write in default constructor (without arguments).
* While initiating the object, based on inputs, if you want to write some logic then use the parameterized constructor.

1. WHAT IS THE DIFFERENCE BETWEEN CONSTRUCTOR AND METHOD?

* While object initiating of the class, constructor will be called and after object initiating the class with the object reference we can call the methods.

1. WHAT IS STATIC AND NON STATIC?

* Non-static variable also known as instance variable while because memory is allocated whenever instance is created.
* Non-static variable are specific to an object.
* Non-static variable can access only with object reference.
* Static variable we can call it as class variables and can access with class reference.
* Static variable not only can be access with class reference but also some time it can be accessed with object reference.
* Static shares the common property to all the objects.
* Even if you create multiple objects in a class it shares the same results for every object.

1. WHAT IS LOCAL VARIABLES?

* Local variables are declared within the methods, constructors, or blocks.
* Access modifiers cannot be used for local variables.
* Local variables are visible only within the declared method, constructor, or block.

1. WHAT IS GLOBAL VARIABLES?

* If we declare any variables outside side of the method, those variables are called as global variables.

WHAT IS SUPER KEYWORD?

* Super keyword refers the immediate parent class methods and variables belongs to object.
* We can access parent class methods and variable like super.variables and
* Super. Methods();
* Also used to invoke parent class constructor from child class

Public Child()

{

Super()

Sysout(“I am child class constructor”)

}

WHAT IS FINAL, FINALLY AND FINALIZE?

FINAL:

* Final is a keyword in java
* We can declare the final for variable’s, method’s, class level.
* If we make any variable, method and class as final.

We cannot modify the final variable, we cannot override the final method and we cannot inherit the final class.

FINALLY {}:

* The finally block is used in association with a [try/catch block](http://www.geeksforgeeks.org/flow-control-in-try-catch-finally-in-java/) and guarantees that a section of code will be executed, even if an exception is thrown. The finally block will be executed after the try and catch blocks.

FINALIZE ():

* It is a method, it calls before the object is garbage collected. It is used for cleanup the garbage collector and it is used to remove some connections like data base connections and browser connections.

1. WHAT IS STRING?

* Strings are immutable in Java it means once created you cannot modify content of String.  It always results in new String.

1. WHAT IS STRING BUFFER?

* String buffer is mutable; we can modify the string variable
* But in the string buffer all the methods are synchronized.

1. WHAT IS STRING BUILDER?

* It is same as string buffer but in the string builder all the methods are not synchronized, we can change the string.

1. WHAT IS STRING TOKENIZER?

* The string Tokenizer class allows a sentence to break a string into tokens.

1. WHAT IS THE PACKAGE IS IMPORTED BY DEFAULT?

Package is collection of classes and interfaces.

Java .Lang package is imported by Default. [Even without package declaration also].

1. WHAT IS AN ARRAY?

* Array is a collection of similar types of data elements
* Array is static in nature and once we assign the array size we cannot modify, it will be constant.
* DataType variable name [] = new DataType[];

**Collections Concept:**

Major difference between Collection and Collections is Collection is an interface and Collections is a class.

1. WHAT IS LIST AND ARRAYLIST AND LINKED LIST?

LIST:

* List is an interface.
* It is used to hold group of objects in single entity.
* It allows duplicate values.
* It follows the insertion order.

ARRAY LIST:

* It’s a class and it implements List interface.
* Array list storing mechanism is similar as Array.
* Array list is very good in case of store and accessing the elements.

LINKED LIST:

* It’s a class and its implements List interface.
* Linked list storage mechanism is Double linked list.
* Linked list is recommended for manipulating (removing/updating) the elements.
* General syntax: List<String> values=new Linked List<String>();

1. WHAT IS SET AND HASHSET?

SET:

* Set is an interface.
* It is used to hold group of objects in single entity.
* It won’t allow duplicate values.
* It will not follow the insertion order.

HASHSET:

* It’s a class and it implements set interface.
* It is used to hold group of objects in single entity.
* It won’t allow duplicate values.
* It will not follow the insertion order.
* General syntax: Set values=new hashSet();

1. WHAT IS MAP AND HASHMAP AND HASHTABLE?

MAP:

* Map is an interface.
* It’s is used to hold group of key and value pairs in single entity.
* It will not allow duplicate keys and it will allow duplicate values.

HASHMAP:

* Hash Map is a class it implements map interface.
* It’s is used to represent group of key and value pairs in single entity.
* It will not allow duplicate keys and it will allow duplicate values.
* Map<Interger, String> m = new HapMap<Integer, String>();
* m.put(key, Value);
* Ex : m.put(206782, "subba");
* It is not synchronized(allows multi threading) and hence not thread safe
* Iterator is used to iterate through the entities

HASHTABLE:

* Hash table is a class, it implements map interface.
* Hash table is legacy class, old implementation.
* It is used to hold group of key and value pairs in single entity.
* Hash table will not allow duplicate keys and values both.
* Main Difference between these two is, in Hash table all methods are synchronized and whereas in Hash Map all the methods are not synchronized.
* Uses Enumerator to iterate through the enity

Hash table is used when code should be more secured and when thread safe i.e thread next thread should wait till other completes

1. WHAT IS EXCEPTION HANDLING AND TYPES?

An exception is an event, which occurs during the execution of a program that disrupts the normal flow of the program's instructions.

1. CHECKD EXCEPTIONS:

* The classes that extend Throwable class except Runtime exceptions and Error are known as checked exceptions
* EX: IO Exception, SQL exception and filenotfindexception etc.
* Checked exceptions we can handle at compile-time.

2. UNCHECKED EXCEPTIONS:

* The classes that extend Runtime exception are known as unchecked exceptions
* EX: ArithmeticException, NullPointerException, ArrayIndexOutOfBoundsException etc.
* Unchecked exceptions are run time exceptions so we cannot handle run time these exceptions at runtime.
* By using try and catch blocks we can handle run time exceptions.

1. WHAT IS TRY AND CATCH BLOCK AND USE?

TRY:

* The try block contains set of statements where an exception can occur.
* A try block is always followed by a catch block, which handles the exception that occurs in associated try block.
* A try block must be followed by catch block or finally block or both.

CATCH:

* A catch block is where you handle the exceptions; this block must follow the try block.
* A single try block can have several catch blocks associated with it. You can catch different exceptions in different catch blocks.
* When an exception occurs in try block, the corresponding catch block that handles that particular exception executes.
* For example if an arithmetic exception occurs in try block then the statements enclosed in catch block for arithmetic exception executes.

1. WHAT IS THROW AND THROWS?

THROW:

* Java throw keyword is used to explicitly throw an exception.
* Throw is followed by an instance.
* Throw is used within the method.
* You cannot throw multiple exceptions.

THROWS:

* Java throws keyword is used to declare an exception.
* Checked exception can be propagated with throws.
* Throws is followed by class.
* Throws is used within the method signature.
* You can declare multiple exceptions
* EX: public void method() throws IOException, SQLException.

1. WHY MAIN METHOD IS STATIC?

* Before starts the execution main JVM will search for main () then it will starts the execution, Static main () will provide the class reference to start execution.
* If it is non-static main () method, JVM will treat it as normal method and it won’t starts the execution.

1. CAN WE OVERLOAD THE STATIC METHOD?

* ‘Yes’. We can have two or more static methods with same name, but differences in input parameters.

1. CAN WE OVERRIDE THE STATIC METHOD?

* Static methods cannot be overridden

1. WHAT IS THIS AND SUPER KEYWORD IN JAVA?

THIS:

It represents the current class and we can call all instance variables and methods using “THIS” keyword.

EX:

* This. Variable;
* This. Method();

R: used to assign local variables to instance variables

SUPER:

* Super keyword refers the immediate parent class methods and variables corresponding to object.
* We can access parent class methods and variable like
* Super. Variables and
* Super. Methods();

1. CAN WE MAKE CONSTRUCTOR AS FINAL?

* We cannot make constructor as final.

1. WHICH CLASS IS THE SUPER CLASS FOR EVERY CLASS?

* object class is the super class of every program [java.lang.Object]

1. WHAT IS GARBAE COLLECTION IN JAVA?
2. WRITE A PROGRAM FOR FACTORIAL?

Class Fact orialExample{

Public static void main (String args[]){

int i,fact=1;

int number=5;//It is the number to calculate factorial.

for(i=1;i<=number;i++){

fact=fact\*i;

}

1. WRITE A PROGRAM FOR FIBONACI SERIES?
2. WRITE A PROGRAM TO REVERSE A STRING WITHOUT USING REVERSE FUNCTION?

Thwowable class is the super class for every call

1. HOW TO READ THE .TXT FILES USING JAVA?

it means if you want to run tests simultaneously in different tabs but in a single browser (instance) window.

Webdriver is not thread-safe. The issue of thread-safety isn't in your code but in the actual browser bindings.

They all assume there will only be one command at a time (e.g. like a real user).

But you can on the other hand instantiate one Webdriver instance for each thread but

it will launch multiple browsers which will consume more memory...

Thread Safe:

A piece of code is thread-safe if it only manipulates shared data structures in a manner that

guarantees safe execution by multiple threads at the same time

2. **import** java.util.ArrayList;
3. **import** java.util.Iterator;
5. **public** **class** IterateThroughArrayListUsingIteratorExample {
7. **public** **static** **void** main(String[] args) {
9. *//create an ArrayList object*
10. ArrayList arrayList = **new** ArrayList();
12. *//Add elements to Arraylist*
13. arrayList.add("1");
14. arrayList.add("2");
15. arrayList.add("3");
16. arrayList.add("4");
17. arrayList.add("5");
19. *//get an Iterator object for ArrayList using iterator() method.*
20. Iterator itr = arrayList.iterator();
22. *//use hasNext() and next() methods of Iterator to iterate through the elements*
23. System.out.println("Iterating through ArrayList elements...");
24. **while**(itr.hasNext())
25. System.out.println(itr.next());
27. }
28. }
30. */\**
31. *Output would be*
32. *Iterating through ArrayList elements...*
33. *1*
34. *2*
35. *3*
36. *4*
37. *5*
38. *\*/*

Access Modifiers

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

1. Default: Variable/Method can be accessed within the package

Ex: void Roopa()

1. Public : Variable/Method can be accessed across all the packages

By creating a object of that class in current class

Package 1

Class A()

{

Public method1()

{

}

}

Package 2

Class B()

{

Psvm()

A a1=new A1();

A1.method1()

}

1. Private : Variable/Method can be accessed only within class
2. Protected : Variable/Method can be accessed only by child class within same package and other packages as well

Programs Tips

1. Convert String to Array and perform logic

char arr[]=Str.toCharArray();

1. Conversion of arraylist to array

Object[] objects = al.toArray();

1. Conversion of array to String

Array.toString();

1. Conversion of array of strings to arraylist

String[] words={“Learn”,”Programming”,”dance”}

List wordlist=Arrays.asList(words);

Programs to prepare:





