Java Interview Q&A

1. What would you rate your Java experience out of 10?

- Best ranges are 6-8 depends on your feeling.
- 6: Not comfortable 7: OK 8: Good
- Don't say 10 or 2-3 either.
- This rating for testing perspective, not from development perspective.

2. What is your Java level? Do you use it in terms of testing only or in terms of development as well?

- Either is correct because every single function you added to the framework considered as Development.
- If you say Development, you can get more coding and data structure questions.
- If you are saying, you are only using for testing you can get less challenging questions.

3. OOP concepts and how you used them in your last project?

- Encapsulation - Inheritance - Abstraction - Polymorphism
- First explain each and then give examples from your Framework.
- Encapsulation is used for hiding the data. Private variables are reached by using public getter and setter methods. Getter is for get the data and Setter is for set the data.
  - Ex: (API Framework) POJO class can be example of Encapsulation because fields will be private and we will able to reach them by public getter and setters.
- Inheritance is used for increasing the reusability because when you inherit one class, you will able to reach that classes fields/features as well.
  - Ex: BasePage class will be inherited by other classes.
  - In the pages directory BasePage class extended by other classes. By that way constructor called by other sub classes. By this way we will able to locate the elements by using the same driver.

- TestBase class is another good example for Inheritance because we are going to use ONLY 1 driver every single test classes.

- TestBase class has several sub-classes.

- Abstraction: There are two ways to achieve abstraction, are abstract class and interface.

  - Why the TestBase and BasePage classes abstract? Because abstract class meant to be inherited and we can not create any object. They meant to be a super class. We don't create any object of both TestBase and BasePage classes therefore they are abstract. We want them to be Inherited by other classes, not creating any OBJECT.

  - It is not mandatory to make TestBase and BasePage classes abstract but making them abstract we can not create any OBJECT of them. They will be ONLY super class.

  - Have you used any interface in your Framework?

  - Yes they are provided by Selenium library such as Alert, Capabilities, JavascpritExecutor, TakeScreenshot, WebDriver etc.

- Also List and Set are Interface as well.
- Can we create an Object from Interface?
- No we can NOT. Interface can be ONLY reference.
- Above interfaces are good examples where we are using abstraction in our Framework.

- When we achieve the abstraction, you will able to create abstract methods which are methods without having any body.

- Polymorphism: Parent can be reference of child object.
  - Ex: Driver class is helping us to achieve multi-browser testing. You will able to test your test cases in different browsers because in the driver class we have the method. The return type of this method is WebDriver therefore even if you are using different driver classes since they are inherited WebDriver, still you will able to get this method by object of different classes such as ChromeDriver, EdgeDriver, FireFoxDriver, etc.
  - Driver class is a good example of Polymorphism in our Framework.

- Ex: List is referenced to ArrayList because List is parent of ArrayList

- Ex: Map is referenced to HashMap and other type of Maps. Or Set —> HashSet

- Whenever when we see parent reference to another object, it will be example of Polymorphism.

4. What do you know about Interface?

- Interface is used to achieve pure abstraction because abstraction is achieved only by abstract class and interface. There is not other way.

- If we have an abstract class why still do we need interface?

- Because 1 class can extend only 1 abstract class but can implement many interfaces.

- We can create such a unique behaviors because if abstract class is inherited, it will have all fields even though were not needed in the child class therefore interface will have only unique features. Therefore if any class needs specific implementations, we can implement related interface.

5. What is the benefit of Interface?

- Main benefit of interface is implementing multiple interfaces.

6. What kind of collections you used in java and your Framework ?

- Collections are one of the 3 Data Structures in Java.
- They are Array, Collections and Map.
- In Framework / Selenium, we can use List of Elements in the dropdown menu to save. If we would like save only the unique ones, then we can use Set.
- In Framework / Selenium, we used Set in order to handle multiple tabs with GetWindowHandles. Its return type is Set because all tabs have a unique ID that is stored in Set data type.

7. Where do you Map in your Framework?

- Map is Key-Value relationship. (Pair of data)
- In my API Framework, at JSON format data stored as Map in the Key-Value format.
- In my UI Framework, ConfigurationProperties contains Key-Value relationship.

8. How to call the last stored variable in an Array?

- variableName.length-1;

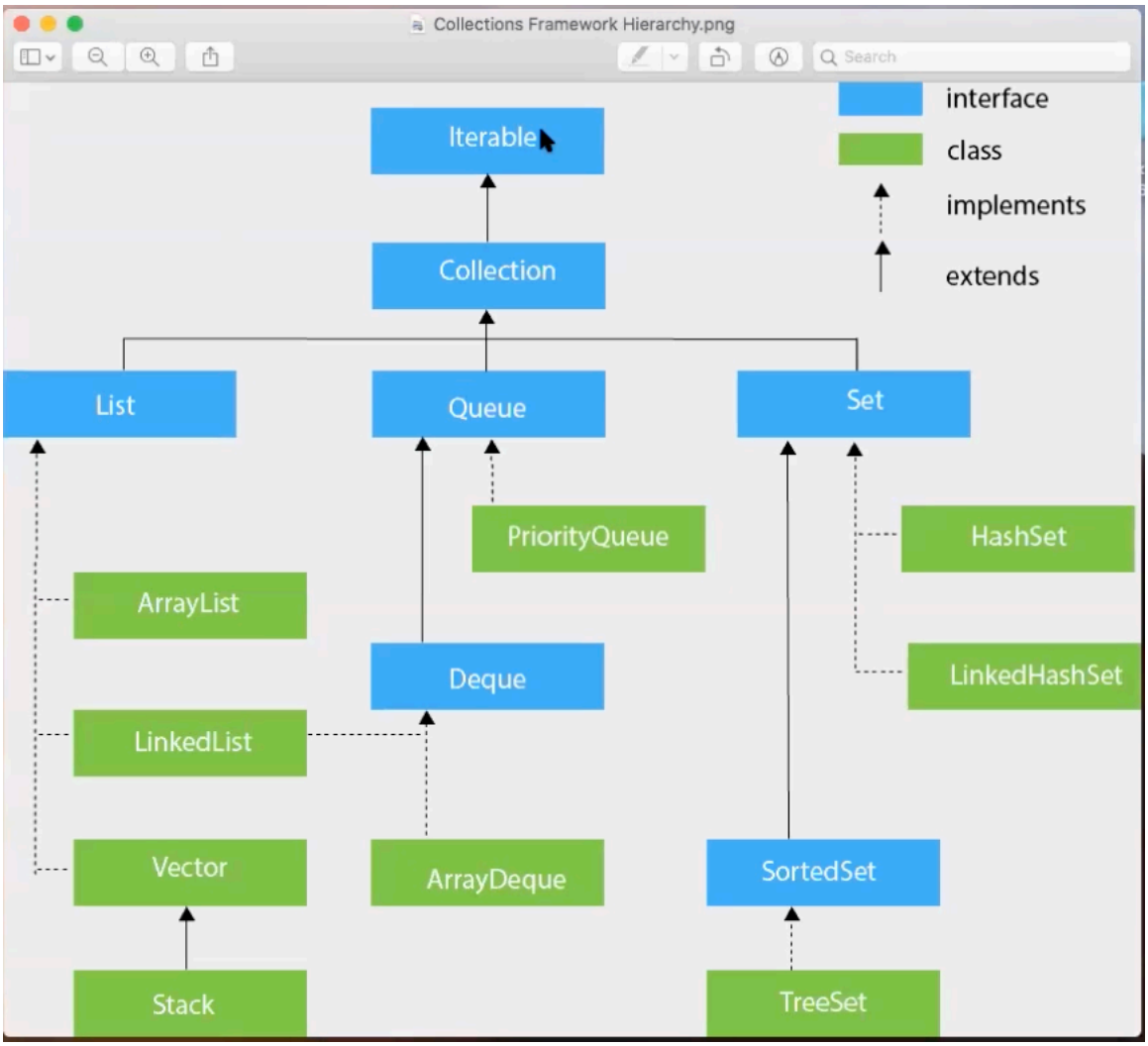
9. Where do you use Set, HashMap, List in your framework?

- Set: Unique options of the dropdown / GetWindowHandles to handle/store multiple tabs
- HashMap: Any place we need to use Maps, we can use HashMap as well. / JSON Data / ConfigurationProperties
- List:

10. What is difference between iterator and for/for each loops?

- When can we use the Iterator? -> The main purpose of Iterator is manipulation the actual object.

- The precondition to use iterator is if the object is implementing Iterable interface. For example we can NOT use Iterator for Array because Array is NOT implementing Iterable interface but we can use Iterable for List, Set, and Queue because they are Iterable interface. Mainly we can say we are using Iterator for Collections because they are implementing Iterable interface.



- For each loop doesn't have any requirement like Iterator, only requirement for the For Each loop is having Data Structure like, Array, Collections and Map.

- When I should use Iterator over For Each loop? A: Another main purpose of Iterator to remove object, if we need to remove object from the Data Structure then Iterator will be a better option.

- Iterate and remove will not be possible with For Each loop therefore we prefer to use Iterator.

- According to the below example, we can only use Iterator, For Each loop will not help to iterate and remove at the same time.

```
public static void main(String[] args) {

    ArrayList<Integer> list = new ArrayList<>(Arrays.asList(100, 100, 100, 200, 200, 300, 300, 400, 100));
    Iterator<Integer> it = list.iterator();
    while (it.hasNext()){
        if(it.next() < 300){
            it.remove();
        }
    }

    System.out.println(list);
```

```
/Library/Java/JavaVirtualMachines/jdk1.8.0_231.jdk/Contents/Home/bin/java ...
[300, 300, 400]

Process finished with exit code 0
```

11. What iterator methods do you use?

- We use the below methods.
- iterator() — hasNext() — next() — remove()

```
Iterator<Integer> it = list.iterator();
while (it.hasNext()){
    if(it.next() < 300){
        it.remove();
```

12. How do you write or read data from or to file with Java?

- In the Framework we use Excel files and we use special library to read and write on it. This library is Apache POI.
- We will also read & write text, CSV files as well.
- If you have a txt file in your project and if you would like to read the information inside this file or write a new information, we can use BufferedReader & BufferedWriter classes without using any external library.

- As seen below example, FileReader will open the file and we need to provide the path of the file in order to find it.

```
// read file with BufferedReader

FileReader file1 = new FileReader( fileName: "TextFile.txt");
BufferedReader read = new BufferedReader(file1);

System.out.println(read.readLine());
System.out.println(read.readLine());
```

- BufferedReader will help us to read the file.

- Also BufferedWriter is used to write on the files.

```
FileWriter file2 = new FileWriter( fileName: "TextFileWrite.txt");
BufferedWriter writer = new BufferedWriter(file2);

writer.write( str: "Hey everyone");

writer.close();
```

- Also we can use Scanner class in order to read files. See the below example. It reads and prints every line of the file.

```
// read file through Scanner

Scanner scan = new Scanner(new FileReader( fileName: "TextFile.txt"));
while(scan.hasNextLine()){
    System.out.println(scan.nextLine());
}
```

- I use Properties class in order to read .properties files.

13. Difference between Heap and Stack?

- Both Heap and Stack are memory locations.
- Heap: is the memory location for all the objects and instances because instances belongs to objects and they stored in the Heap memory.
- Stack: is the memory location any local variables. Local variables are any variable declared in a method, included main method.

14. You have a list of fruits which some of them repeats. How would you write a method which will return a fruit that repeats the most?

- First I find the frequency of the every single fruit

- Then whichever fruit has highest frequency then it is the fruit needs to be returned.

15. Java Version you are using?

- I am using Java version 8.

- You can say 11 as well but don't say anything lower than 8. Also 9 and 12 are outdated, you can not download them anymore.

16. Abstraction vs interface ?

- Classes can be abstract and the features we have in class like Constructor, instance methods, variables etc but Interface doesn't have them.

- The advantage of interface over abstract class is multiple inheritance.

- The access modifiers we use interface is only public. We can not use other public modifiers. But in abstract class we can use any access modifiers.

- Abstract class is meant to be super class, meant to be extended

17. Java generics ?

- Java generics means the types like int, double, boolean etc. It means Type-Safety.

- Num = 100; we can not declare it because data type was not clarified.

- According to the below example l1 variable will accept any data type because acceptable data type didn't mentioned anywhere, therefore it will accept int, String, boolean, etc.

```
56        List l1 = new ArrayList();
57
58        l1.addAll( Arrays.asList(1, "A", 'C', false));
59
60        System.out.println(l1);
61
62
63
64
```

```
Review
How are you all doing?
I am good how about
============================
[1, A, C, false]
```

- If I try to assign String value on the above example, it will give me NumberFormatException.

18. What is HashMap explain, how do use in your framework?

- Pair of data and whenever we are using Map in our framework we can use as HashMap like JSON, Properties, POJO class, serialization, deserialization, etc.

19. What is thread safe? How do you make your code thread safe?

- Thread-Safe means one thread at a time.
- Thread exist every single application.
- Sub-sequence of the process called Thread and exist in every single process. 1 process can use multiple/many threads.
- How can we make it Thread-Safe in Java? A: To achieve in Java, we use special keyword which is called synchronized keyword. We can use synchronized block or method to make it thread safe.

20. String builder/buffer differences?

- String - StringBuilder - StringBuffer : All of the for char sequences in Java.
- String: Immutable char sequence in Java.

- String Builder: mutable char sequence in Java.
- String Buffer: mutable char sequence in Java.
- Immutable means you can not change, can't be modified.
- StringBuffer is synchronized but not StringBuilder therefore StringBuffer is slower than StringBuilder.

```
String s1 = new String( original: "A");

// mutable:
StringBuilder s2 = new StringBuilder("B");
StringBuffer s3 = new StringBuffer("C");  // synchronized
```

21. How can you achieve abstraction?

- 2 ways to achieve -> abstract class and interface

22. What is the difference btw List, Queue and Set?

- List accept duplicates but Set doesn't.
- List have index number but Set doesn't have any index number.
- Queue accept duplicates but doesn't have any index number.
- What makes Queue is special is it has a special order which is FIFO.
- The opposite is Stack which is LIFO.

- Stack is the sub-class of List.

23. What is Map?

- Covered before.

24. How do you iterate over HashMap?

- If you need to iterate only values, you need to use values() method.
- If you need to iterate only key, you need to use keySet() method.
- If you need to iterate both key and values, then you need to create EntrySet() method

25. What is garbage?

- Garbage means the object that is not going to be use anymore. And Garbages are eligible for Garbage Collection.
- This process is done implicitly. finalize() method is called by Java to collect unreferenced values.
- According to the this example true is eligible for Garbage Collection.

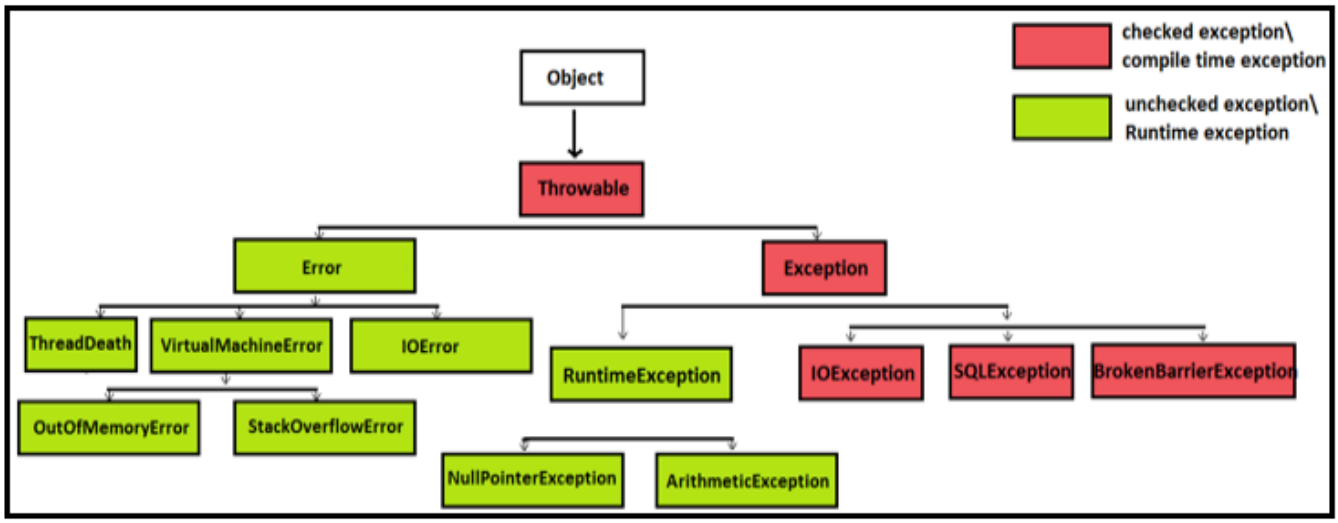
```
Boolean b1 = true;
        b1 = null;
```

26. What is a class in java?

- Class is the template of the object. When you have a class, you will able to create objects of that class.
- You can create as many objects of that class.
- We can not create object without the class because objects are coming from the class.

27. What kind of exception you face and how do you handle them?

- What are the exceptions in Java?



- RuntimeException
- NoSuchMethodException
- ClassCastException

- ArrayIndexOutOfBoundException
- ArithmeticException: 10/0
- NullPointerException: The reason to create an object to be able to reach instances.
- Below image explain the reason of the exception. Below String object referring null therefore we will not able to use String class methods.
- toLowerCase() is the instance of the String object but String object is referring null therefore it will give NullPointerException.
- As of now, r1 is null, not an object.
- Whenever object is null and you are trying to call its instances, it will give NullPointerException

```
String r1 = null;
System.out.println( r1.toLowerCase() );
```

```
Review
---------------------------
[1]
[]
Exception in thread "main" java.lang.NullPointerException Create breakpoint
    at reviewSection.Review.main(Review.java:85)
```

- What are the exceptions in Selenium?

- Selenium has ONLY unchecked exceptions that we see during runtime.
    - NoSuchElementException
    - NoSuchFrameException
    - NoSuchFrameException
    - ElementNotVisibleException
    - TimeOutException
    - StaleElementException

28. What is the difference between a HashMap and HashTable?

- HashMap: is not synchronized. Accept null keyword.
- HashTable: is synchronized. Doesn't accept null keyword.

29. What is polymorphism? What is runtime polymorphism?

- Polymorphism focus on objects and behaviors of the objects.
- Runtime polymorphism is the same thing with method overriding.

```
// WebDriver driver =new ChromeDriver();
// diver.get(URL)
```

According to the this example ChromeDriver() get method will be executed because it is the overriding one.

- In polymorphism always overriding one executed.

30. Give me an example from your framework where you used OOP concepts.

- See Question 3.

31. What is global variable?

- Static variable is called global variable because it is acceptance. It is accepted any place.
- For example if you have a static variable, you can use it inside the instance variable but other way is not possible.
- In order to static you don't need any object, you can call by the class name.
- Instance variables are not called Global because they were not accepted inside the static methods.

32. How do you handle exceptions? What is the most recent exception that you get?

- See Question 27
- If you need to handle you can use try & catch block or throws exception to the method signature

33. Difference between protected and default?

- protected: visible in the same package and also visible in inherited classes on different package.
- default: visible only in the same package.

34. Can you give "public" access modifier where you are overriding a protected method ?

- Yes because public is more visible than protected.
- During overriding rule is access modifier should be same or more visible.
- We can not override private methods because they are only visible in the same class but in order to override we need to have in different class. And private is not visible outside the class.
- We can not override final, because it is the final version but override require to implement new things.
- We can not override static because static belong to class and it will have only one copy.

35. How can you find max value in an int array?

- You can use loop and assign the first element to max and then compare with others, if there are larger one and then update the max_value variable.

36. Can class be final?

- Yes class can be final. if it is final then it is not going to be inherited and will not have child class. Super class can not be final.
- Abstract class can not be final but other classes can be final.
- final class: can not have a child class.
- abstract class: can not create an object of that class.
- A class can NOT be abstract and final at the same time.
- final variable can not be re-assigned and final method can not be overriding.

37. Static methods?

- Static methods belong to the class and has only one copy.
- You can call through the class.

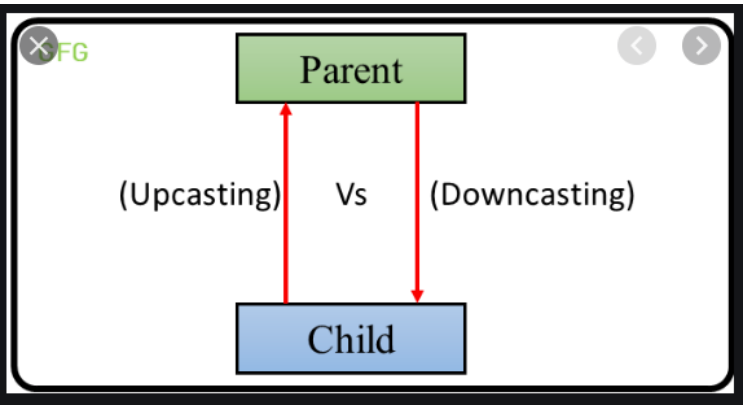
- I used static methods in my utilities package to call them by class name.

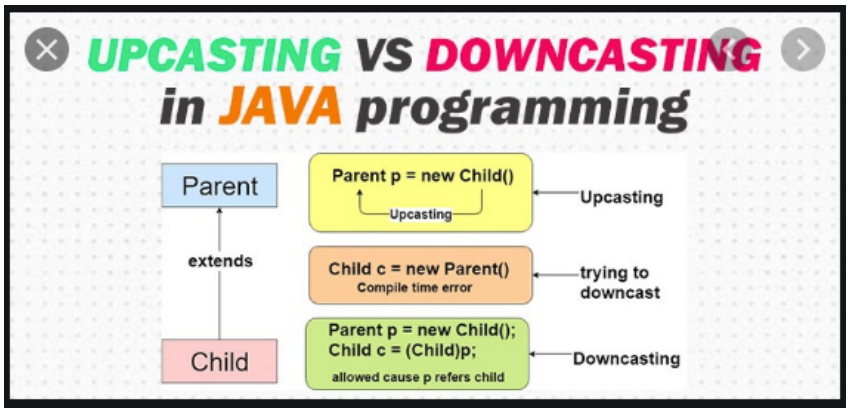
38. non-static methods?

- Instance methods are belong to the object. We need to create an object in order to call these methods.

39. Up casting, down casting?

- Casting the reference type. Part of Polymorphism, casting one reference type to another.
- Up-casting: Casting from small type to larger. Up-casting is automatically done, you don't need to do anything.
- Down-casting: Casting from larger to smaller. The reason of down-casting is calling the methods from the object type.





40. What is the difference between final, finally and finalize?

- final is a keyword and you can apply to class, method and variables.

- finally is a block and we use after the try&catch and regardless exception fixed or not run the finally block.
- finalize is a method to collect garbages and called by Java implicitly.

41. Tell me how do you understand immutability in java and what is the difference between mutable?

- Immutability means once you created the object you can NOT change it. Mutable means you can change it after object is created.
- String is immutable class and String class is final. By using final keyword we can make classes immutable.
- If any class include final keyword, you can NOT inherit as well.
- If any class is immutable, it will be much faster.

42. Can we have multiple inheritance in java? If yes, how do we achieve that?

- For classes NO, for interface YES and we achieve it by implement keyword.

43. How did you implement polymorphism in your Framework?

- We use in Driver class. We make WebDriver as reference and then create object of ChromeDriver, FirefoxDriver, EdgeDriver, etc.

```
Integer a = null;

a = a + 4;

System.out.println("a = " + a);

// gives => NullPointerException
```

44. What could be the reason if you are getting NullPointerException

- null keyword doesn't reference any object.
- Primitives doesn't accept null but classes accepts.

45. What are the main differences between Array - ArrayList?

- Both are data structures.
- Array: Accepts both primitives and objects and has index number. Has a fixed size therefore you cn not add and remove new elements.
- ArrayList: Accept only objects and has index number. Size is dynamic, it can change by adding and removing new elements.
- Array is faster than ArrayList.

46. Method overloading VS Method Overriding?

- Method Overloading is Compile time Polymorphism.
- Method Overriding is Runtime Polymorphism.
- Overriding has ONLY 1 method but different implementations.
- Overloading has different methods with the same name and different parameters or different number of parameters.
- Overriding happens in the sub class but Overloading can happen in the same class or different classes.
- Overriding return type, must be same and access modifier must be same or more visible. Parameters must be same as well because we are talking about the same method with different type of implementation.
- The ONLY rule of Overloading is parameters. Parameters must be different. Other access modifier, return type doesn't matter same or different because according to the parameters different methods will be called.
- Any method you created, you can OVERLOAD.
- final, static, private methods can not be OVERRIDED.
- abstract methods can be overrided because abstract methods are meant to be overrided.

47. What are the differences between Throw and Throws keyword?

- Both of them are doing opposite functions. One creates exception other handles the exception.
- Throw: is used to create exception.
- Throws: is used to handle exceptions.
- Throws keyword is used inside method signature to handle exception tentatively for checked exceptions.

48. Tell me the differences between HashMap and LinkedHashMap?

- Both accept null keyword. They were not synchronized
- HashMap: It doesn't keep the insertion order but it is the fastest one compared to others.
- LinkedHashMap: Keeps the insertion order and they are tinkly connected each other, therefore adding and removing process is faster.

49. Tell me the differences between Static and Instance Variable/Block?

- Answered earlier.

50. What's the use of super keyword? What's the use this keyword?

- super keyword: reference object of super class.
- this keyword: reference object of this/current class.
- this. => instance variables and methods of this class
- super. => instance variables and methods of the super class
- this() => Constructor of this class.
- super() => Constructor of super class.
- We need Constructor to create an object from the class.
- When we are creating a new object, we use new keyword and then Constructor of that class.
- We use the super keyword inside the sub class to declare super class methods and Constructor.

51. Why OOP concept is important for producing a software?

- What is the main reason of the OOP is reusing the object, reducing the code.
- By applying the OOP, you don't need to repeat same codes again and again. You are going to make an object re-usable.
- Re-using the code makes it easy to maintain and looks cleaner.