Assignment 4

1. What's the difference between final, finally? What is finalize()?

The final, finally, and finalize are keywords in Java that are used in exception handling. Each of these keywords has a different functionality. The basic difference between final, finally and finalize is that the final is an access modifier, finally is the block in Exception Handling and finalize is the method of object class.

2. What's the difference between throw and throws?

S. No.	Key Difference	throw	throws
1.	Point of Usage	The **throw** keyword is used inside a function. It is used when it is required to throw an Exception logically.	The **throws** keyword is used in the function signature. It is used when the function has some statements that can lead to exceptions.
2.	Exceptions Thrown	The **throw** keyword is used to throw an exception explicitly. It can throw only one exception at a time.	The **throws** keyword can be used to declare multiple exceptions, separated by a comma. Whichever exception occurs, if matched with the declared ones, is thrown automatically then.
3.	Syntax	Syntax of **throw** keyword includes the instance of the Exception to be thrown. Syntax wise throw keyword is followed by the instance variable.	Syntax of **throws** keyword includes the class names of the Exceptions to be thrown. Syntax wise throws keyword is followed by exception class names.
4.	Propagation of Exceptions	**throw** keyword cannot propagate checked exceptions. It is only used to propagate the unchecked Exceptions that are not checked using the throws keyword.	**throws** keyword is used to propagate the checked Exceptions only.

3. What are the two types of exceptions?

Checked and Unchecked

4. What is error in java?

In Java, an error is a subclass of Throwable that tells that something serious problem is existing, and a reasonable Java application should not try to catch that error. Generally, it has been noticed that most of the occurring errors are abnormal conditions and cannot be resolved by normal conditions.

5. Exception is object, true or false?

True

6. Can a finally block exist with a try block but without a catch?

Yes, it is possible to have a try block without a catch block by using a final block. As we know, a final block will always execute even there is an exception occurred in a try block, except System.

7. From java 1.7, give an example of the try-resource feature.

Prior to Java SE 7, you can use a `finally` block to ensure that a resource is closed regardless of whether the `try` statement completes normally or abruptly. The following example uses a `finally` block instead of a `try`-with-resources statement:

```
static String readFirstLineFromFileWithFinallyBlock(String path)
                                                    throws IOException {
    BufferedReader br = new BufferedReader(new FileReader(path));
    try {
        return br.readLine();
    } finally {
        br.close();
    }
}
```

8. What will happen to the Exception object after exception handling?

The Exception object will be garbage collected in the next garbage collection.

9. Can we use String as a condition in switch(str){} clause?

Yes

10. What's the difference between ArrayList, LinkedList and vector?

ArrayList	**LinkedList**
It uses a dynamic array as it's internal implementation.	It uses doubly linked list as it's internal implementation.
It is better in get and set operations.	It is better in adding and removing operations.

ArrayList	**Vector**
It is not synchronized.	It is synchronized.
It is not a legacy class.	It is a legacy class.
It increases its size by 50% of the array size.	It increases its size by doubling the array size i.e. 100%.
Iterator interface is used to traverse the ArrayList elements.	Iterator or Enumeration interface can be used to traverse the Vector elements.

ArrayList is much fast than Vector because it is non-synchronized.	Vector is slow as compared ArrayList because it is synchronized

11. What's the difference between hashTable and hashMap?

S. No.	Hashmap	Hashtable
1.	No method is synchronized.	Every method is synchronized.
2.	Multiple threads can operate simultaneously and hence hashmap's object is not thread-safe.	At a time only one thread is allowed to operate the Hashtable's object. Hence it is thread-safe.
3.	Threads are not required to wait and hence relatively performance is high.	It increases the waiting time of the thread and hence performance is low.
4.	Null is allowed for both key and value.	Null is not allowed for both key and value. Otherwise, we will get a null pointer exception.
5.	It is introduced in the 1.2 version.	It is introduced in the 1.0 version.
6.	It is non-legacy.	It is a legacy.

12. What is static import?

In Java, static import concept is introduced in 1.5 version. With the help of static import, we can access the static members of a class directly without class name or any object. For Example: we always use sqrt() method of Math class by using Math class i.e. Math.sqrt(), but by using static import we can access sqrt() method directly.

According to SUN microSystem, it will improve the code readability and enhance coding. But according to the programming experts, it will lead to confusion and not good for programming. If there is no specific requirement then we should not go for static import.

13. What is static block?

In a Java class, a static block is a set of instructions that is run only once when a class is loaded into memory. A static block is also called a static initialization block. This is because it is an option for initializing or setting up the class at run-time.

14. Explain the keywords: default(java 1.8), break, continue, synchronized, strictfp, transient, volatile, instanceOf

Break: Java break keyword is used to break the loop or switch statement. It breaks the current flow of the program at specified conditions.

Continue: Java continue keyword is used to continue the loop. It continues the current flow of the program and skips the remaining code at the specified condition.

Synchronized: Java synchronized keyword is used to specify the critical sections or methods in multithreaded code.

Strictfp: Java strictfp is used to restrict the floating-point calculations to ensure portability.

Transient: Java transient keyword is used in serialization. If you define any data member as transient, it will not be serialized.

Volatile: Java volatile keyword is used to indicate that a variable may change asynchronously.

instanceOf: Java instanceof keyword is used to test whether the object is an instance of the specified class or implements an interface.

15. Create a program including two threads – thread read and thread write.

Input file ->Thread read -> Calculate -> buffered area

Buffered area -> Thread write -> output file

Detailed description is in assignment4.txt file.

Sample input.txt file. Attached files are input.txt and a more detailed description file.

ReaderThread will read from the input file.

```
import java.io.BufferedReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.util.concurrent.BlockingQueue;

public class ReaderThread implements Runnable{

 protected BlockingQueue<String> blockingQueue = null;

 public ReaderThread(BlockingQueue<String> blockingQueue){
  this.blockingQueue = blockingQueue;
 }

 @Override
 public void run() {
  BufferedReader br = null;
   try {
        br = new BufferedReader(new FileReader(new File("./input.txt")));
        String buffer =null;
        while((buffer=br.readLine())!=null){
           blockingQueue.put(buffer);
        }
        blockingQueue.put("EOF");  //When end of file has been reached

     } catch (FileNotFoundException e) {
```

```
            e.printStackTrace();
        } catch (IOException e) {

            e.printStackTrace();
        } catch(InterruptedException e){

        }finally{
            try {
                br.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }


    }
```

WriterThread will write to output file.

```
import java.io.File;
import java.io.FileNotFoundException;
import java.io.PrintWriter;
import java.util.concurrent.BlockingQueue;

public class WriterThread implements Runnable{

  protected BlockingQueue<String> blockingQueue = null;

  public WriterThread(BlockingQueue<String> blockingQueue){
    this.blockingQueue = blockingQueue;
  }

  @Override
  public void run() {
    PrintWriter writer = null;

    try {
      writer = new PrintWriter(new File("output.txt"));

      while(true){
        String buffer = blockingQueue.take();
        //Check whether end of file has been reached
        if(buffer.equals("EOF")){
          break;
        }
        writer.println(buffer);
      }


    } catch (FileNotFoundException e) {

      e.printStackTrace();
    } catch(InterruptedException e){

    }finally{
      writer.close();
```

```
    }

  }

}


import java.util.concurrent.ArrayBlockingQueue;
import java.util.concurrent.BlockingQueue;

public class Launcher {

  public static void main(String[] args) {

    BlockingQueue<String> queue = new ArrayBlockingQueue<String>(1024);

    ReaderThread reader = new ReaderThread(queue);
    WriterThread writer = new WriterThread(queue);

    new Thread(reader).start();
    new Thread(writer).start();

  }

 }
```