Digital Assignment - 2
Name: KULVIR SINGH
Reg. No.:19BCE2074
Slot: G1
Course: CSE1007 – JAVA PROGRAMMING

Question 1 :

Write a java program to count the occurrence of a particular word in a file. Assume that the files are text files. Write each file name, along with the number of occurrences in that file, to standard output. If an error occurs while trying to read from the files, you should print an error message for that file.

Answer:

```
import java.io.*;
import java.util.*;

class word_occurence{
   static int wordCount(String x,String w)throws IOException{
      File f1 = new File(x);
      FileReader fr = new FileReader(f1);
      BufferedReader br = new BufferedReader(fr);
      String s;
      int i;int flag =0;
      String words[]=null;
      while((s=br.readLine())!= null){
         words = s.split(" ");
         for(i = 0;i<words.length;i++){
            if(words[i].equalsIgnoreCase(w))
```

```
            flag++;
          }
       }
       br.close();
       fr.close();
       return flag;
   }
   public static void main(String[] args)throws IOException {
       Scanner x = new Scanner(System.in);
       String in;int i=0;
       String directory = "C:/Users/kulvir/Desktop/JAVA CODes/files";
       File f = new File(directory);
       String content[] = f.list();
       for(i=0;i<content.length;i++){
          System.out.println("Enter the word to be found in file "+content[i]);
          in=x.next();
          System.out.println("Occurences of "+in+" in the text file "+content[i]+" = "+wordCount(content[i],
in));
       }
   }


}
```

Question 2:

Implement the following scenario using Java

A main thread is first starting a thread and later it's stopping that thread by calling our stop() method, which uses a boolean volatile variable to stop running thread e.g. Server. The Server, which implements Runnable is running fine until main() method called the stop() method, the only then value of boolean volatile variable exit is set to true. In next iteration, the Server thread checks and find this value true, so it come out of the loop and run() method, which means your thread is now stopped.

Answer:

```
class MyThread implements Runnable {
        private boolean exit;

        private String name;
        Thread t;

        MyThread(String threadname)
        {
                name = threadname;
                t = new Thread(this, name);
                System.out.println("New thread: " + t);
                exit = false;
                t.start();
        }

        public void run()
        {
```

```
		int i = 0;
		while (!exit) {
			System.out.println(name + ": " + i);
			i++;
			try {
				Thread.sleep(100);
			}
			catch (InterruptedException e) {
				System.out.println("Caught:" + e);
			}
		}
		System.out.println(name + " Stopped.");
	}
	public void stop()
	{
		exit = true;
	}
}


public class thread {
	public static void main(String args[])
	{
		MyThread t1 = new MyThread("First thread");
		MyThread t2 = new MyThread("Second thread");
		try {
			Thread.sleep(500);
			t1.stop();
			t2.stop();
```

```
                Thread.sleep(500);
            }
            catch (InterruptedException e) {
                System.out.println("Caught:" + e);
            }
            System.out.println("Exiting the main Thread");
        }
}
```

Question 3:

List and explain the exception handling keywords in java. Also write about the exception hierarchy

Answer:

The Exception Handling in Java is one of the powerful mechanisms to handle the runtime errors so that normal flow of the application can be maintained.

The exception handling keywords are:

1. try
2. catch
3. throw
4. throws
5. finally

try

Java try block is used to enclose the code that might throw an exception. It must be used within the method. If an exception occurs at the particular statement of try block, the rest of the block code will not execute. So, it is recommended not to keeping the code in try block that will not throw an exception. Java try block must be followed by either catch or finally block.

catch

Java catch block is used to handle the Exception by declaring the type of exception within the parameter. The declared exception must be the parent class exception (i.e., Exception) or the generated exception type. However, the good approach is to declare the generated type of exception. The catch block must be used after the try block only. You can use multiple catch block with a single try block.

throw

The Java throw keyword is used to explicitly throw an exception. We can throw either checked or unchecked exception in java by throw keyword. The throw keyword is mainly used to throw custom exception. We will see custom exceptions later. The syntax of java throw keyword is given below.

throws

The Java throws keyword is used to declare an exception. It gives an information to the programmer that there may occur an exception so it is better for the programmer to provide the exception handling code so that normal flow can be maintained. Exception Handling is mainly used to handle the checked exceptions. If there occurs any unchecked exception such as NullPointerException, it is programmers' fault that he is not performing checkup before the code being used.

finally

Java finally block is a block that is used *to* execute important code such as closing connection, stream etc. Java finally block is always executed whether exception is handled or not. Java finally block follows try or catch block.