5COSC001W - Solutions to Tutorial 8 Exercises

1 How Exceptions change the flow of control of your program

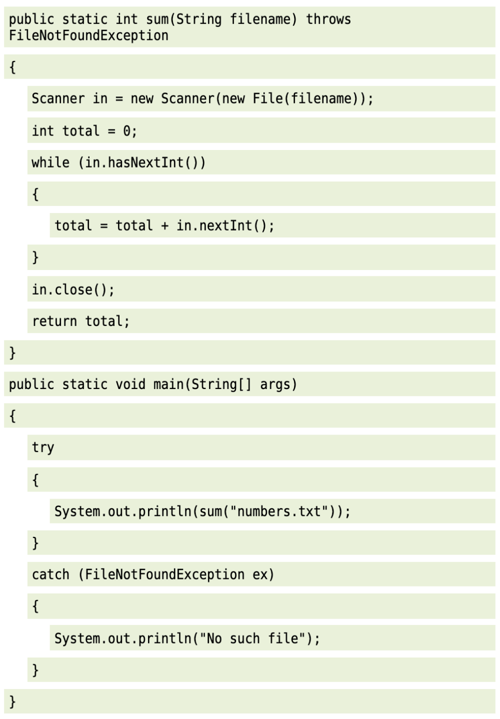
Try it out! Ask the help of your tutor if you cannot make it after working on it for quite some time!

2 File Opening Exception

Try it out! Ask the help of your tutor if you cannot make it after working on it for quite some time or you cannot understand why you got the correct/incorrect answer!

3 Throwing Exceptions

Make sure the order and indentation is right! See Figure 1 for the correct solution.

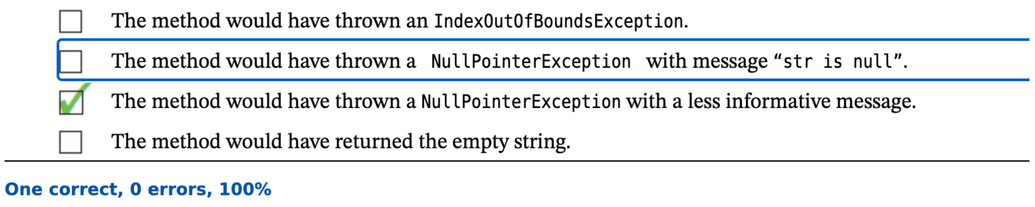


```
public static int sum(String filename) throws
FileNotFoundException
{
    Scanner in = new Scanner(new File(filename));
    int total = 0;
    while (in.hasNextInt())
    {
        total = total + in.nextInt();
    }
    in.close();
    return total;
}
public static void main(String[] args)
{
    try
    {
        System.out.println(sum("numbers.txt"));
    }
    catch (FileNotFoundException ex)
    {
        System.out.println("No such file");
    }
}
```

Figure 1: Question 3 solution.

4 Throwing Exceptions (cont'ed)

See Figure 2 for the correct solution.



- [ ] The method would have thrown an `IndexOutOfBoundsException`.
- [ ] The method would have thrown a `NullPointerException` with message "`str is null`".
- [✓] The method would have thrown a `NullPointerException` with a less informative message.
- [ ] The method would have returned the empty string.

One correct, 0 errors, 100%

Figure 2: Question 4 solution.

5 More on Learning to Deal with Exceptions

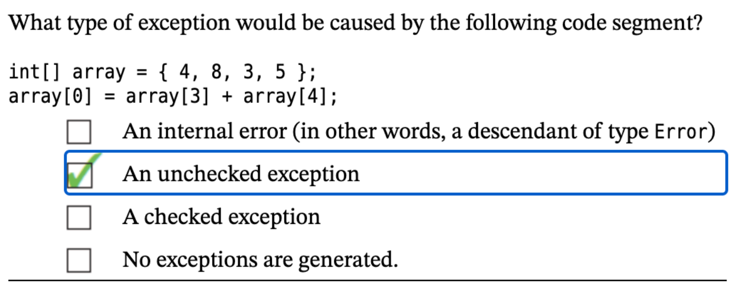
Make sure the order and indentation is right! See Figure 3 For correct solution.

```
public static int sum(String filename) throws
FileNotFoundException
{
    Scanner in = new Scanner(new File(filename));
    int total = 0;
    while (in.hasNextInt())
    {
        total = total + in.nextInt();
    }
    in.close();
    return total;
}
public static void main(String[] args)
{
    try
    {
        System.out.println(sum("numbers.txt"));
    }
    catch (FileNotFoundException ex)
    {
        System.out.println("No such file");
    }
}
```

Figure 3: Question 5 solution.

6 Testing your Knowledge on the Types of Exceptions

Make sure the order and indentation is right! See Figure 4 For correct solution.



What type of exception would be caused by the following code segment?

```
int[] array = { 4, 8, 3, 5 };
array[0] = array[3] + array[4];
```

- ☐ An internal error (in other words, a descendant of type `Error`)
- [✓ An unchecked exception]
- ☐ A checked exception
- ☐ No exceptions are generated.

Figure 4: Question 6 solution.

7 Dealing with Exceptions

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

public class ListOfNumbers {
    private List<Integer> list;
    private static final int SIZE = 10;

    public ListOfNumbers () {
        list = new ArrayList<Integer>(SIZE);
        for (int i = 0; i < SIZE; i++)
            list.add(new Integer(i));
    }

    public void writeList() {
        PrintWriter out = null;

        try {
            System.out.println("Entering try statement");
            out = new PrintWriter(new FileWriter("OutFile.txt"));

            for (int i = 0; i < list.size(); i++)
                out.println("Value at: " + i + " = " + list.get(i));
        } catch (IndexOutOfBoundsException e) {
            System.err.println("Caught IndexOutOfBoundsException: " +
                               e.getMessage());
        } catch (IOException e) {
            System.err.println("Caught IOException: " + e.getMessage());
        } finally {
            if (out != null) {
                System.out.println("Closing PrintWriter");
                out.close();
            } else {
                System.out.println("PrintWriter not open");
            }
        }
    }

    public void readList() {
        File fp = new File("InFile.txt");

        /* try with resources; Scanner implements autocloseable
           therefore this is possible and scanner will close
           automatically */
        try (Scanner sc = new Scanner(fp)) {
            while (sc.hasNext()) {
                int i = sc.nextInt();
                list.add(i);
                System.out.println(i);
            }
        }
        catch (FileNotFoundException ex) {
            System.err.println("Exception: " + ex);
        }
    }


    public static void main(String[] args) {
        ListOfNumbers program = new ListOfNumbers();
        program.readList();
        program.writeList();

    }
}
```

The input file `InFile.txt` needs to be created:

```
1223 111 554
0 12 1171771 676 7799
161616 1799 0001 67616
161616717
```

8 More on Exceptions

```
import java.io.*;

class ExceptionsExampleQ2 {
    public static void cat(File file) {
        RandomAccessFile input = null;
        String line = null;

        try {
            input = new RandomAccessFile(file, "r");
            while ((line = input.readLine()) != null) {
                System.out.println(line);
            }
            return;
        }
        catch (FileNotFoundException ex) {
            ex.printStackTrace();
        }
        catch (IOException ex) {
            ex.printStackTrace();
        }
        finally {
            if (input != null) {
                try {
                    input.close();
                }
                catch (IOException ex) {
                    ex.printStackTrace();
                }
            }
        }
    }

    public static void main(String[] args) {
        ExceptionsExampleQ2 program = new ExceptionsExampleQ2();
        program.cat(new File("filename"));
    }
}
```