CS164: Exception Worksheet
Name(s):_____

1. Is the following code legal? Explain.

```
try {

} finally {

}
```

2. What exception types can be caught by the following handler?

```
catch (Exception e) {

}
```

What is the problem with using this type of exception handler?

3. Is there anything wrong with the following exception handler as written? Will this code compile? Explain.

```
try {

} catch (Exception e) {

} catch (ArithmeticException a) {

}
```

4. The following program demonstrates a runtime unchecked exception that is caused by dividing a number by zero. Correct this code, by adding the appropriate exception handling, so the program can function correctly. Tip: Arithmetic abnormalities like divide by zero results in ArithmeticException.

```
public class ArithmeticExceptionExample {
   public static void main(String args[]) {
            int num1=10;
            int num2=0;
            //divide both numbers and print the result
            int result=num1/num2;
            System.out.println(result);
   }
}
```

5. The following program demonstrates a runtime unchecked exception. Correct this code, by adding the appropriate exception handling, so the program can function correctly. Tip: NullPointerException occurs when you try to access methods over an object that has null reference.

```
public class NullPointerExceptionExample {
    public static void main(String args[]){
        String str = null;
        System.out.println(str.length());
    }
}
```

6. Explain line by line what is happening in the code below. Explain in which situations this code will work or not work.

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

public class Main {
    public static void findFile() throws IOException {
```

```
        File newFile = new File("text.txt");
        Scanner stream = new Scanner(newFile);
        while(stream.hasNext()){
            System.out.println(stream.nextLine());
        }
        stream.close();
    }

    public static void main(String[] args) {
        try {
            findFile();
        }
        catch (IOException e) {
            System.out.println(e);
        }
    }
}
```

7. Write a new method for the FilesException Class that reads inputs (Strings) from the user and write those inputs into a file. The method ends when the user enter "exit". First think about what do you need to do to solve this problem (divide-glue-conquer). Write a sequence of steps in English on how to do that. Translate your sequence of steps to a Java method.

```
import java.io.File;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.Scanner;

public class FilesException {

    private String fileName;

    public FilesException(String fileName){
        this.fileName = fileName;
    }

    public String readFile(){
        String strFromFile = "";
        Scanner scnrFile = null;
        try{
            scnrFile = new Scanner(new File(fileName));
            while(scnrFile.hasNext()){
                strFromFile += scnrFile.nextLine() + "\n";
            }
        }catch(FileNotFoundException fileExp){
            System.out.println("File not found!");
        }catch(IOException ioExp){
            System.out.println("Something wrong with file!");
        }finally {
            scnrFile.close();
        }
        return strFromFile;
    }
}
```