CS 111, Programming Fundamentals II
Lab 7: Exceptions

Computer Science

This lab is meant to further your understanding of exceptions, and give you hands-on experience with try-catch blocks.

I. Introduction

We've discussed in lecture, how to catch exceptions, so that your program exits 'gracefully'. Catching exceptions also allows you to write programs that generate an appropriate response, based on the error that is thrown. In this lab you are given a buggy code, to which you will add try-catch blocks, so that the program does not crash when it encounters an error. To get started download the following files from Canvas:

- ***ABuggyProgram.java***
- ***Car.java***
- ***myDataFile.txt***

Open the java file and explore it, have a look at the switch and the methods being called inside of the cases. Compile and run the program. Right now the code does not handle any exceptions, when you call any of the methods, the program will crash and print out the stack. The java program ***ABuggyProgram*** has four intentional errors. Your goal is to catch those errors, so that the program does not crash.

II. Handling Errors of type *FileNotFoundException*

You will add a try-catch clause to the method named ***openFile()***. First run the code and select the option number 1, the program will crash. Explore the stack printed out, notice the line where the program encounter a problem. The error message will tell you which exception has been generated, you can check the Java API to get a better understanding about the specific exception. Do the following to handle this exception.

1. Wrap all of the code inside of the method named ***openFile()*** with a try clause. Have a look at code snipped below, all of the code in the method should be part of the "try code block".

2. Add a single catch statement. The exception type being caught must match that one being throw inside of this method. The exception should be ***FileNotFoundException***, and the reference variable name can be just about anything you want. Inside of the "catch code block" add a System.out.println() with the message "****File can't be open, it does not exist.****". You can also print out the message generated by the exception, by calling the method ***getMessage()*** on the reference variable of the exception. Add another print statement and call the method inside of it.

3. Remove the "*throws FileNotFoundException*" from the method header, compile and run the program. The program should not crash anymore when option number 1 is selected and the method named ***openFile()*** is called.

```
try{
   //try code block
}catch(ExceptionType referenceVariable){
   //catch code block
}
```

III. Handling Errors of type *NumberFormatException*

Run the code and select the option number 2, the program will crash. Explore the stack printed out, notice the line where the program encounter a problem. The error message will tell you which exception has been generated, you can check the Java API to get a better understanding about the specific exception. Do the following to handle this exception.

1. Similarly to the last part of the lab, put all of the code inside of the method named ***parseNumber(),*** into a try clause.

2. Add a single catch statement. The exception being caught should be ***NumberFormatException***, and the reference variable name can be just about anything you want. Inside of the "catch code block" add a System.out.println() with the message "****A conversion error happened****".

3. Compile and run the program. The program should not crash anymore when option number 2 is selected and the method named ***parseNumber()*** is called.

IV. Handling Errors of type *InputMismatchException*

Run the code and select the option number 3, the program will crash. Explore the stack printed out, notice the line where the program encounter a problem.

1. Similarly to the last part of the lab, put all of the code inside of the method named ***readPoorlyFormattedFile(),*** into a try clause.

2. Inside of this method, there will be two separate exception which you will need to catch Have a look at the code snippet below. First add a catch statement which can catch an exception of type ***FileNotFoundException***. Inside of the "catch code block" add a System.out.println() with the message "****File can't be open, it does not exist.****". Exactly like you did in the previous part of this lab.

3. Secondly, add a catch statement which can catch an exception of type ***InputMismatchException***. Inside of the "catch code block" for this catch clause add a System.out.println() statement. In the print statement put the message "****Input was incorrect****".

4. Compile and run the program. The program should not crash anymore when option number 3 is selected and the method named ***readPoorlyFormattedFile ()*** is called.

```
try{
   //try code block
}catch(ExceptionType e1){
   //catch 1 code block
   System.out.println(e1.getMessage());
}catch(ExceptionType e2){
   //catch 2 code block
}
```

V. Handling Errors of type *NullPointerException*

Run the code and select the option number 4, the program will crash. Explore the stack printed out, notice the line where the program encounter a problem. This time you will not handle this exception inside of the method where the exception is being throw. Instead you will handle this exception inside of the calling method (main method).

1. Add a try-catch inside of the switch case in the main method. In case number 4, modify the code in such way that the method named ***createCarArray()*** which is being called, is part of the "try code block".

2. Add a single catch statement. The exception being caught should be ***NullPointerException***. Inside of the "catch code block" add a System.out.println() with the message "****The array has NOT been populated with car objects****".

3. The entire try-catch needs to be inside of the case 4 of the switch. The break statement should be after the closing curly bracket of the catch clause. Compile and run the program. The program should not crash anymore when option number 4 is selected and the method named ***createCarArray ()*** is called.

VI. Main method *InputMismatchException*

After you have taken care of all of the exceptions run the program and enter a character or a word instead of a number. An exception should be generated. Have a look at the stack trace and find out which line you have to wrap in a try-catch clause. In the catch statement assign a random number to the ***userInput***, any number that's not 0-4.

VII. Sample Output

Sample output of the program can be seen below on **Figure 1**.

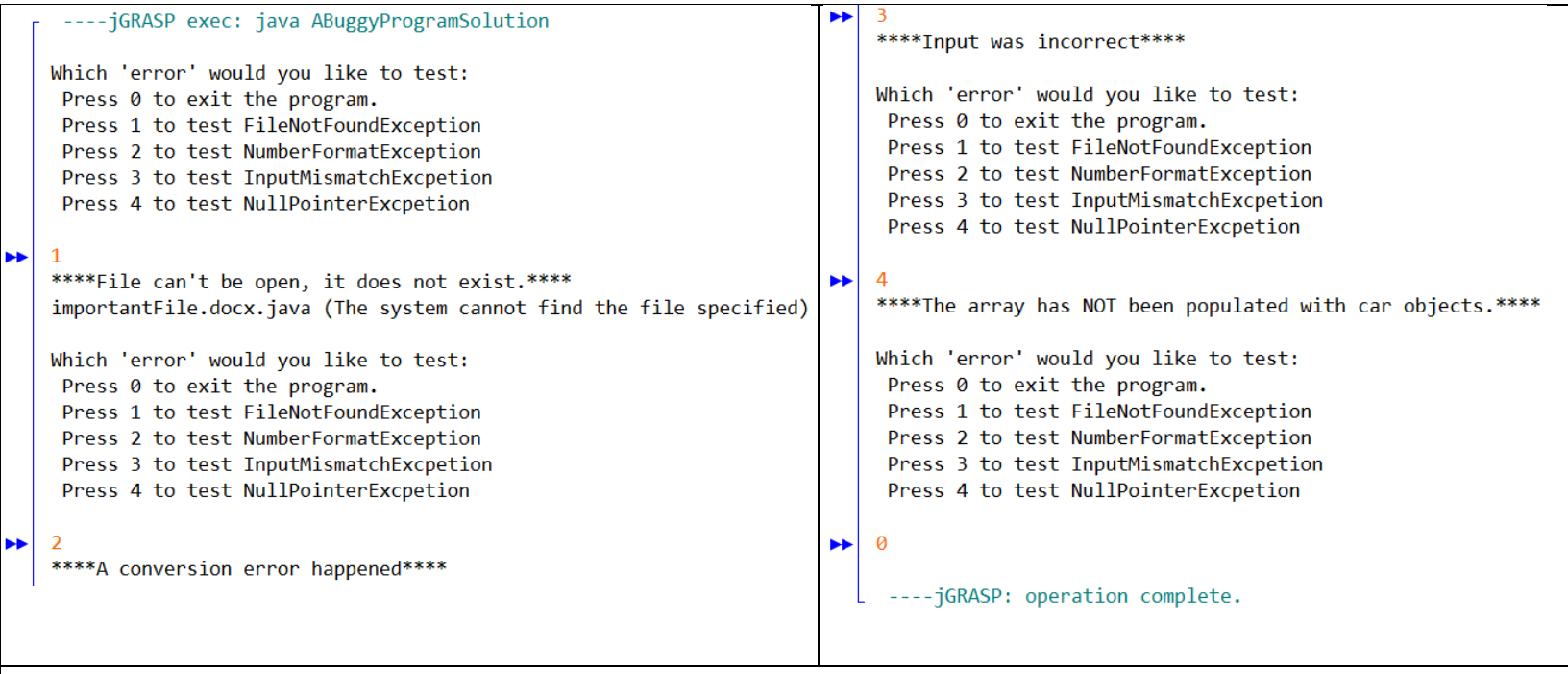

Figure 1 : Sample output

VIII. Upload your work to Canvas

For this lab, make sure that you upload the following file to the Lab 7 assignment in your Canvas account:

ABuggyProgram.java

Rubric

File / Lab	Points
Exception *FileNotFoundException* is handled inside of the ***openFile()*** method	20
Exception *NumberFormatException* is handled inside of the ***parseNumber()*** method	20
Two exceptions are handled inside of the method ***readPoorlyFormattedFile()***	20
Exception *NullPointerException* is handled inside of the switch in ***main()*** method	20
Exception *InputMismatchException* is handled inside of the ***main*** method	10
Code compiles and runs as expected	5
Code is indented and commented	5
Total	100