COP 3330 Suggested Exercises for Week 12

Exceptions

1) The first program shown during the semester, Circle2.java, would crash if the user entered a non-integer radius. Edit this program so that if the user enters the wrong type of data (InputMismatchException) that you catch this Exception and reprompt the user to re-enter the radius, and continue to do so until the user enters an integer. Here is a link to the program:

https://www.cs.ucf.edu/~dmarino/ucf/introjava/Circle2.java

2) Another early program shown in class was one that reads in test scores and computes their average. Though unlikely, if all scores entered the user are invalid, then no average exists. In the original version an if statement is used to avoid the division by 0. Rewrite this code so that the division is attempted in a try clause and the ArithmeticException is caught so that the program doesn't crash by dividing by 0. Here is the original program to edit:

https://www.cs.ucf.edu/~dmarino/ucf/introjava/Average.java

3) Write a program that asks the user to enter a string and an index into the string. Try printing out the character stored at that index, but if the index is out of bounds, catch the StringIndexOutOfBoundsException and print out an error message stating that the string doesn't have a character at that index.

4) Write code that utilizes objects in some way shape or form and catches a potential NullPointerException.

5) Write code that utilizes arrays in some way and catches a potential ArrayIndexOutOfBoundsException.

File Input

6) The following directory stores four files related to a programming contest problem (the description, test input file, matching output, and solution). The solution was coded reading input from standard input and it was tested on the command prompt utilizing file redirection. Rewrite the code to directly read from the file "sorting.in" and produce the output to standard output. Here is the directory:

https://www.cs.ucf.edu/~dmarino/ucf/introjava/SortingStudent/

7) Update your solution to #6 so that it catches a FileNotFoundException, if the file to be opened isn't in the current directory. In this case, print a message stating the fact and exit the program.

8) Write a program that reads in a dictionary of words from the file dictionary.txt. The format of the file is that the first line contains a single positive integer representing the number of words, then the words follow, one per line. After reading in the dictionary into TreeMap, as the user to enter a word to search for. If it's in the dictionary, say so, if it's not in the dictionary, then print out the closest word alphabetically after the string entered by the user and the closest word that comes alphabetically before the string entered by the user (if these exist).

9) Modify #8 to play a simple word game that uses the dictionary to check for validity of words. (One possibility is the game where users go back and forth listing valid words where the next word must start with the letter the last word ended with.)

10) Find data online that is in a text file format and write a program to process that data in some way.