Week8Lab – 10 pts

Pre-lab questions

1. Why would we want to catch an exception?

Java exception handling is important because it helps maintain the normal, desired flow of the program even when unexpected events occur. If Java exceptions are not handled, programs may crash, or requests may fail.

You should catch the exception when you are in the method that knows what to do. For example, you are writing a library for opening and reading files. Here, the programmer knows what to do, so they catch the exception and handle it.

1. What might your program output when catching an IndexOutOfBoundsException?

The IndexOutOfBoundsException is thrown when attempting to access an invalid index within a collection, such as an array, vector , string , and so forth. It can also be implemented within custom classes to indicate invalid access was attempted for a collection.

The program which I use for next question in that it will print Error = 0 when catching an IndexOutOfBoundException.

1. Where does a ‘throws’ clause go? Write an example.

Throw clause will go into the catch block where it will catch by the catch keyword.

public class Mars {

public int Setname(String Input) {

int output = 0;

try {

output = Input.indexOf("L");

System.out.println("Error:" + Input.indexOf("L"));

if(output == -1){

throw new IndexOutOfBoundsException();

}

}

catch (IndexOutOfBoundsException e) {

System.out.println("Error:2" + e.getStackTrace());

}

return output;

}

public static void main(String[] args) {

Mars O = new Mars();

O.Setname("London");

}

}

Choose one of the following to develop into a program that will allow you to practice with search and sort. Once chosen, do the following:

Understand the problem (restate in your own words, make any assumptions clear):

I choose the Temperature Averages program in that I am creating array to store the temp values in it. then finding their highest lowest and average value. After that I will ask user to print for the start and end value. Then I have to insert those values in array to print from start to end and again find the lowest highest and average value of that array. I use try catch block if the value that does not in array it will print exception caught.

Pseudocode:

Scanner class for asking user input for the number of temp values in array. Then start and end value. Then I use two arrays one for normal temp array second is for start to end value array for temp. using try catch for the for loops where the exception going to happen, I use try catch. General flow for the first array taking input from user then printing the temp array after that finding the min max and avg. same for second array which start from start and end at end value. Then min max and avg for that array.   
  
Name of file (.java) submitted:

Temp.java

Temperature Averages

Create an array of ints that represent temperatures. Find the highest and lowest temperature, print them out then calculate the range. Calculate and print the average temp for the timeframe represented by the array. Ask the user to select a range of temperatures by entering two ints, then give them the average in that span. Ex. User inputs 10 and 20, your program outputs the average temp for those 10 values. Since we can’t control users, put this in a try/catch block in case they enter a number that is not an array index.

Gas prices

Create an array of doubles that represent gas prices. Find the highest and lowest price, print them out then calculate the range. Then sort the list in ascending order and find the middle value, print that out. Ask the user to select a gas price by entering a double and use binary search to find that value in the list. Tell the user where it is in the list and how much lower the lowest value is and how much higher the highest value is. Since we can’t control users, put this in a try/catch block in case they enter a number that is not in the array.

Dice rolls

Create an array of ints that represent the roll of a 6 sided die. Populate an array with 1000 simulated die rolls. Find the first roll for each number and print it out (ex. 1 @ index 3, 2 @ index 0, etc) using linear search. Sort the list and count how many of each roll there are (ex. 1 rolled 152 times, 2 rolled 166 times, etc). Shuffle the list then ask the user for an int between 0 and 999 and return the die roll at that location. Put this in a try/catch loop just in case.