Student Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Class and Section \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Total Points (40 pts + n extra pts) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Due: Refer to Blackboard (Late homework submission is not acceptable! You must turn in your own work in blackboard!)**

#### **Project: Project: Calculator**

CS210 Introduction to Programming Principles

California Lutheran University

Problem Description:

Develop a Calculator that takes inputs from the user for the following operations (+, -, \*, /, =) as well as numbers to be computed.

Sample 1:

Enter the problem statement:

2 + 4 \* 5 = 22

Click AC to restart or Exit to stop.

You must have one method for each one of those operations.

For more sample your can use the online [calculator](https://www.google.com/search?safe=active&rlz=1C1CHBF_enUS729US729&sxsrf=ALeKk00R8WkMOnlhE-HJKfZt3aSn8s-rNQ%3A1584502077331&ei=PZVxXu_iE4zY-wT-3KbgDg&q=Calculator&oq=Calculator&gs_l=psy-ab.12..0i71l8.0.0..16953...0.3..0.0.0.......0......gws-wiz.s-pkYE1rMkI&ved=0ahUKEwiv9dubiqPoAhUM7J4KHX6uCewQ4dUDCAs).

Submit the following items:

1. Submit this world document with your answers before the due date.
2. Zip your project and submit it before the due date.
3. Analysis: (10 pts)

(Describe the program use cases. Write down the requirements for your code. Describe the stakeholders for your code. Describe what the SW should be able to do?)

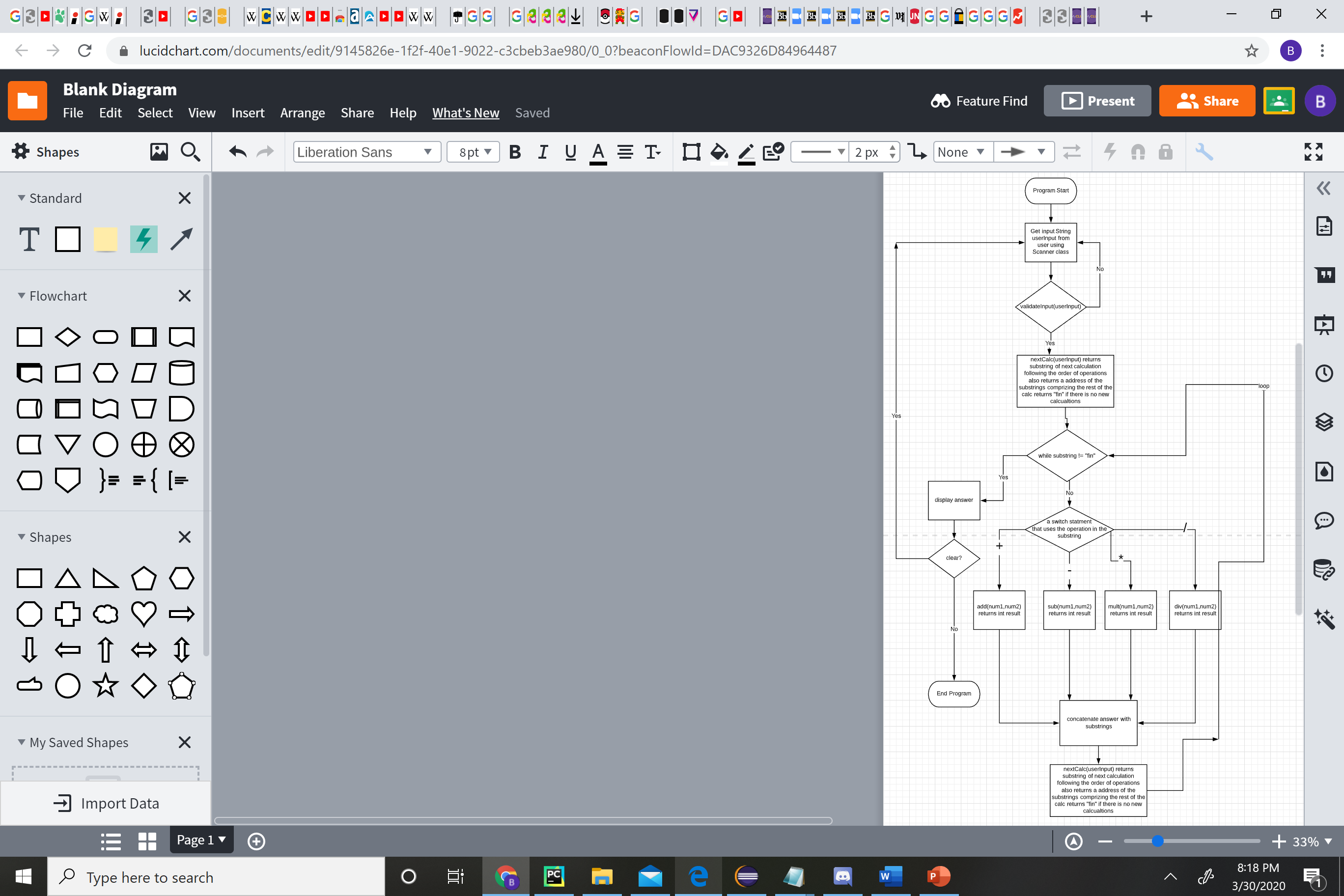
* Scope/Objective:

Design a Calculator Software in JAVA that takes inputs from the user for the following operations (+, -, \*, /, =) as well as numbers to be computed.

* Use cases:
  + Simple math calculations
  + Elementary school kids
  + People who are doing quick math
* Requirements:
  + The SW shall be written in JAVA.
  + The SW shall check the user input and make sure a valid input has been entered and will prompt the user to re-enter.
  + The SW shall not crash.
  + The SW shall not have any coding errors.
  + The SW shall be documented well.
  + The SW shall be readable.
  + The SW shall be testable.
  + The Software shall user the scanner to take the user input as a string data type.
  + The Software will evaluate the string as follows
    - Search String for first instance of \* or /
    - If no instance of the above look for first instance of + and -
    - Save address -1 and address+1 in an int
    - Use address to make a substring and substring before and after
    - Figure out what method will neeSyd to be called in terms of calculation type.
    - Cast string to double and put numbers from substring into method
    - Save answer.
    - Concatenate answer with before and after substring
    - Repeat until answer remains
  + The SW shall display the answer and then ask if the user wants to clear or continue.

1. Design: (10 pts)

(Describe the major steps for solving the problem. Only a flow chart will be acceptable.)



1. Coding: (10 pts)

(Copy and Paste Source Code here. Format your code using Courier 10pts. Screen shot of your code is also acceptable.)

package cs210;

import java.util.ArrayList;

import java.util.Scanner;

import java.math.\*;

public abstract class HW7 {

/\*

\* }else if((Character.toString(userInput.charAt(i+1)) == "!")) { double fire =

\* (double)Integer.parseInt(s); double total = 1; for(int x = (int) fire; 0 < x;

\* x--) { total = total \* x; } nums.add(fire); }else {

\*

\* double fire = (double)Integer.parseInt(s); nums.add(fire);

\*

\*

\* }

\*/

public static void main(String[] args) {

String s = "";

String z = "";

String userInput;

Scanner newSc = new Scanner(System.in);

// Opening

System.out.println("Welcome to Ben's Calculator");

boolean con = false;

boolean con3 = false;

ArrayList<String> operations = new ArrayList<String>();

ArrayList<Double> nums = new ArrayList<Double>();

while (con3 == false) {

con = false;

while (con == false) {

System.out.println(

"Please insert calculation as a line of text seperated by operation symbols (+ - \* / ^x) and type = when your ready to calcualte");

// Input

userInput = newSc.next();

for (int i = 0; i <= userInput.length() - 1; i++) {

//

try {

if (Character.toString(userInput.charAt(i)).contains("+")

|| Character.toString(userInput.charAt(i)).contains("-")

|| Character.toString(userInput.charAt(i)).contains("/")

|| Character.toString(userInput.charAt(i)).contains("\*")

|| Character.toString(userInput.charAt(i)).contains("=")

|| Character.toString(userInput.charAt(i)).contains("^")

|| Character.toString(userInput.charAt(i)).contains("s")) {

if ((Character.toString(userInput.charAt(i)).contains("^"))) {

i = i + 1;

while (Character.toString(userInput.charAt(i)).contains("0")

|| Character.toString(userInput.charAt(i)).contains("1")

|| Character.toString(userInput.charAt(i)).contains("2")

|| Character.toString(userInput.charAt(i)).contains("3")

|| Character.toString(userInput.charAt(i)).contains("4")

|| Character.toString(userInput.charAt(i)).contains("5")

|| Character.toString(userInput.charAt(i)).contains("6")

|| Character.toString(userInput.charAt(i)).contains("7")

|| Character.toString(userInput.charAt(i)).contains("8")

|| Character.toString(userInput.charAt(i)).contains("9")) {

String d = Character.toString(userInput.charAt(i));

int y = Integer.parseInt(d);

z = z + Integer.toString(y);

i = i + 1;

}

double fire1 = Math.pow((double) Integer.parseInt(s), (double) Integer.parseInt(z));

nums.add(fire1);

} /\*

\* else if(Character.toString(userInput.charAt(i)).contains("s")) { i = i + 1;

\* if(Character.toString(userInput.charAt(i)).contains("q")) { i = i + 1;

\* if(Character.toString(userInput.charAt(i)).contains("r")) { i = i + 1;

\* if(Character.toString(userInput.charAt(i)).contains("t")) { i = i + 1;

\* while(Character.toString(userInput.charAt(i)).contains("0")||Character.

\* toString(userInput.charAt(i)).contains("1")||Character.toString(userInput.

\* charAt(i)).contains("2")||Character.toString(userInput.charAt(i)).contains(

\* "3")||Character.toString(userInput.charAt(i)).contains("4")||Character.

\* toString(userInput.charAt(i)).contains("5")||Character.toString(userInput.

\* charAt(i)).contains("6")||Character.toString(userInput.charAt(i)).contains(

\* "7")||Character.toString(userInput.charAt(i)).contains("8")||Character.

\* toString(userInput.charAt(i)).contains("9")) { String d =

\* Character.toString(userInput.charAt(i)); int y = Integer.parseInt(d); z = z +

\* Integer.toString(y); i = i + 1; } double fire2 =

\* Math.sqrt((double)Integer.parseInt(s)); nums.add(fire2); } }

\*

\* } }

\*/else {

double fire = (double) Integer.parseInt(s);

nums.add(fire);

}

s = "";

z = "";

operations.add(Character.toString(userInput.charAt(i)));

if (Character.toString(userInput.charAt(i)).contains("=")) {

con = true;

break;

}

} else {

String f = Character.toString(userInput.charAt(i));

int a = Integer.parseInt(f);

s = s + Integer.toString(a);

}

} catch (Exception e) {

System.out.println("Invalid Input");

nums.clear();

operations.clear();

break;

}

}

}

ArrayList<String> numCalc = new ArrayList<String>();

double result;

while (!(operations.get(0).equals("="))) {

numCalc = nextCalc(numCalc, operations, nums);

switch (numCalc.get(0)) {

case "+":

result = addition(Double.valueOf(numCalc.get(1)), Double.valueOf(numCalc.get(2)));

nums.set(Integer.valueOf(numCalc.get(3)), result);

int sa = Integer.valueOf(numCalc.get(4));

nums.remove(sa);

int sb = Integer.valueOf(numCalc.get(3));

operations.remove(sb);

numCalc.clear();

break;

case "-":

result = subtraction(Double.valueOf(numCalc.get(1)), Double.valueOf(numCalc.get(2)));

nums.set(Integer.valueOf(numCalc.get(3)), result);

int sc = Integer.valueOf(numCalc.get(4));

nums.remove(sc);

int sd = Integer.valueOf(numCalc.get(3));

operations.remove(sd);

numCalc.clear();

break;

case "\*":

result = multiplication(Double.valueOf(numCalc.get(1)), Double.valueOf(numCalc.get(2)));

nums.set(Integer.valueOf(numCalc.get(3)), result);

int se = Integer.valueOf(numCalc.get(4));

nums.remove(se);

int sq = Integer.valueOf(numCalc.get(3));

operations.remove(sq);

numCalc.clear();

break;

case "/":

result = division(Double.valueOf(numCalc.get(1)), Double.valueOf(numCalc.get(2)));

nums.set(Integer.valueOf(numCalc.get(3)), result);

int sw = Integer.valueOf(numCalc.get(4));

nums.remove(sw);

int st = Integer.valueOf(numCalc.get(3));

operations.remove(st);

numCalc.clear();

break;

case "=":

break;

}

}

System.out.println(nums.get(0));

boolean con2 = false;

while (con2 == false) {

System.out.println("Type AC to restart or EXIT to stop.");

String userResponse = newSc.next();

userResponse = userResponse.toUpperCase();

if (userResponse.contains("AC")) {

operations.clear();

nums.clear();

break;

} else if (userResponse.contains("EXIT")) {

con2 = true;

con3 = true;

System.out.println("Goodbye, Have a Nice Day!");

} else {

System.out.println("Invalid Input");

}

}

}

}

private static double division(Double valueOf, Double valueOf2) {

return ((double) valueOf / valueOf2);

}

private static double multiplication(Double valueOf, Double valueOf2) {

return ((double) valueOf \* valueOf2);

}

private static double subtraction(Double valueOf, Double valueOf2) {

return ((double) valueOf - valueOf2);

}

private static double addition(Double valueOf, Double valueOf2) {

return ((double) valueOf + valueOf2);

}

public static ArrayList<String> nextCalc(ArrayList<String> numCalc, ArrayList<String> operations,

ArrayList<Double> nums) {

for (int i = 0; i <= operations.size() - 1; i++) {

if (operations.get(i).equals("\*") || operations.get(i).equals("/")) {

if (operations.get(i).equals("\*")) {

numCalc.add("\*");

}

if (operations.get(i).equals("/")) {

numCalc.add("/");

}

numCalc.add(String.valueOf(nums.get(i)));

numCalc.add(String.valueOf(nums.get(i + 1)));

numCalc.add(String.valueOf((i)));

numCalc.add(String.valueOf((i + 1)));

return numCalc;

}

}

for (int i = 0; i <= operations.size() - 1; i++) {

if (operations.get(i).equals("+") || operations.get(i).equals("-")) {

if (operations.get(i).equals("+")) {

numCalc.add("+");

}

if (operations.get(i).equals("-")) {

numCalc.add("-");

}

numCalc.add(String.valueOf(nums.get(i)));

numCalc.add(String.valueOf(nums.get(i + 1)));

numCalc.add(String.valueOf((i)));

numCalc.add(String.valueOf((i + 1)));

return numCalc;

}

}

for (int i = 0; i <= operations.size() - 1; i++) {

if (operations.get(i).equals("=")) {

numCalc.add("=");

return numCalc;

}

}

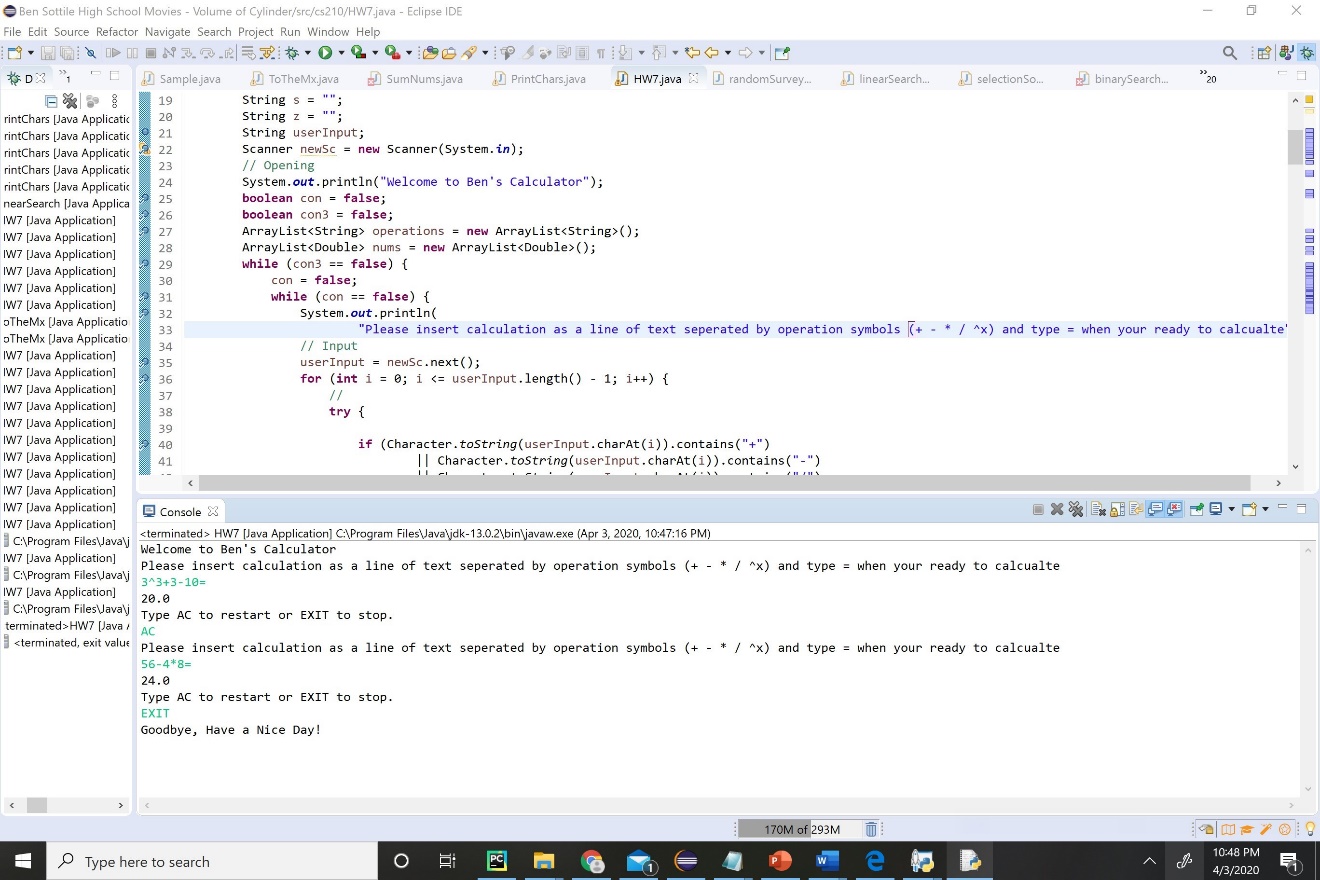
return null;

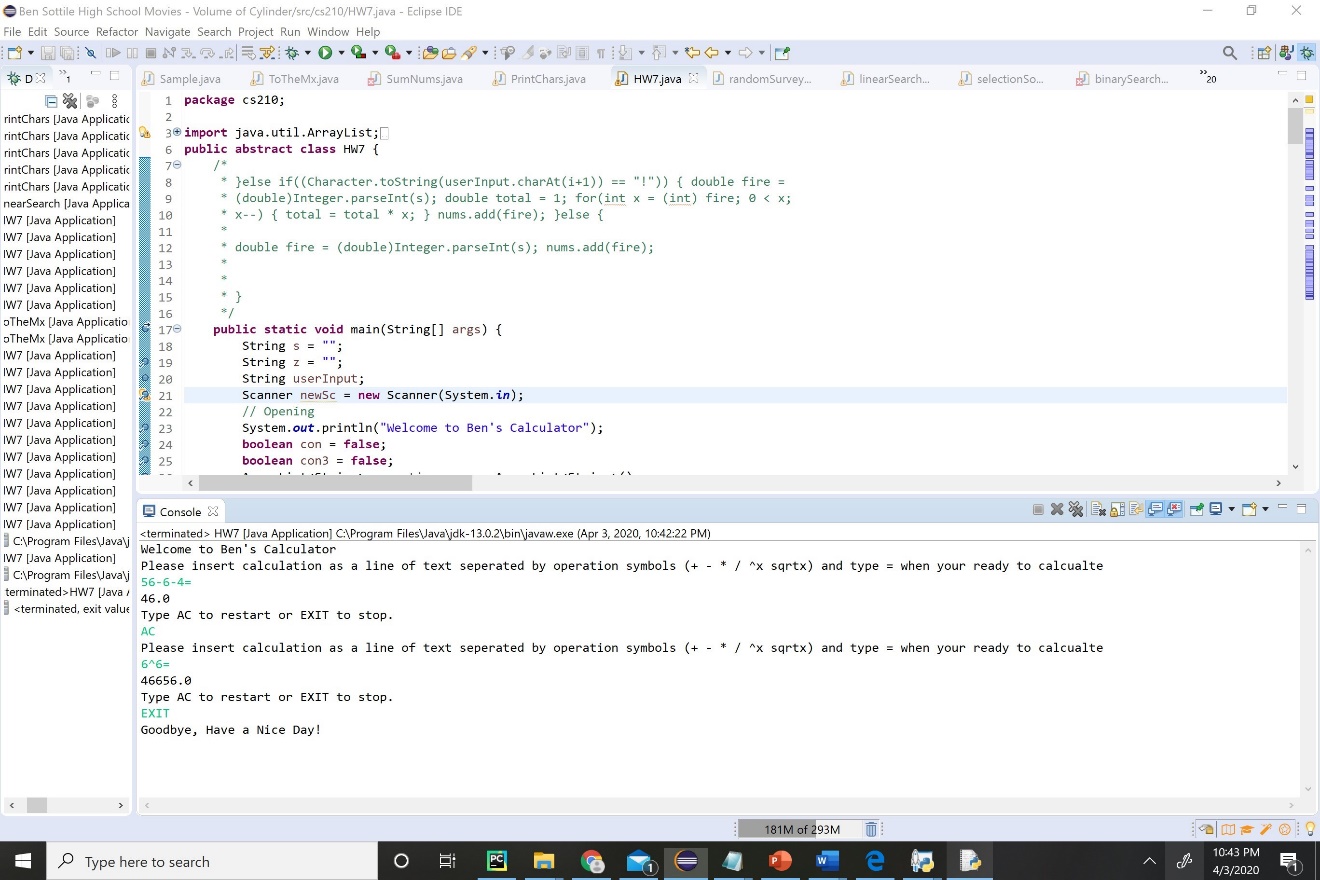
}

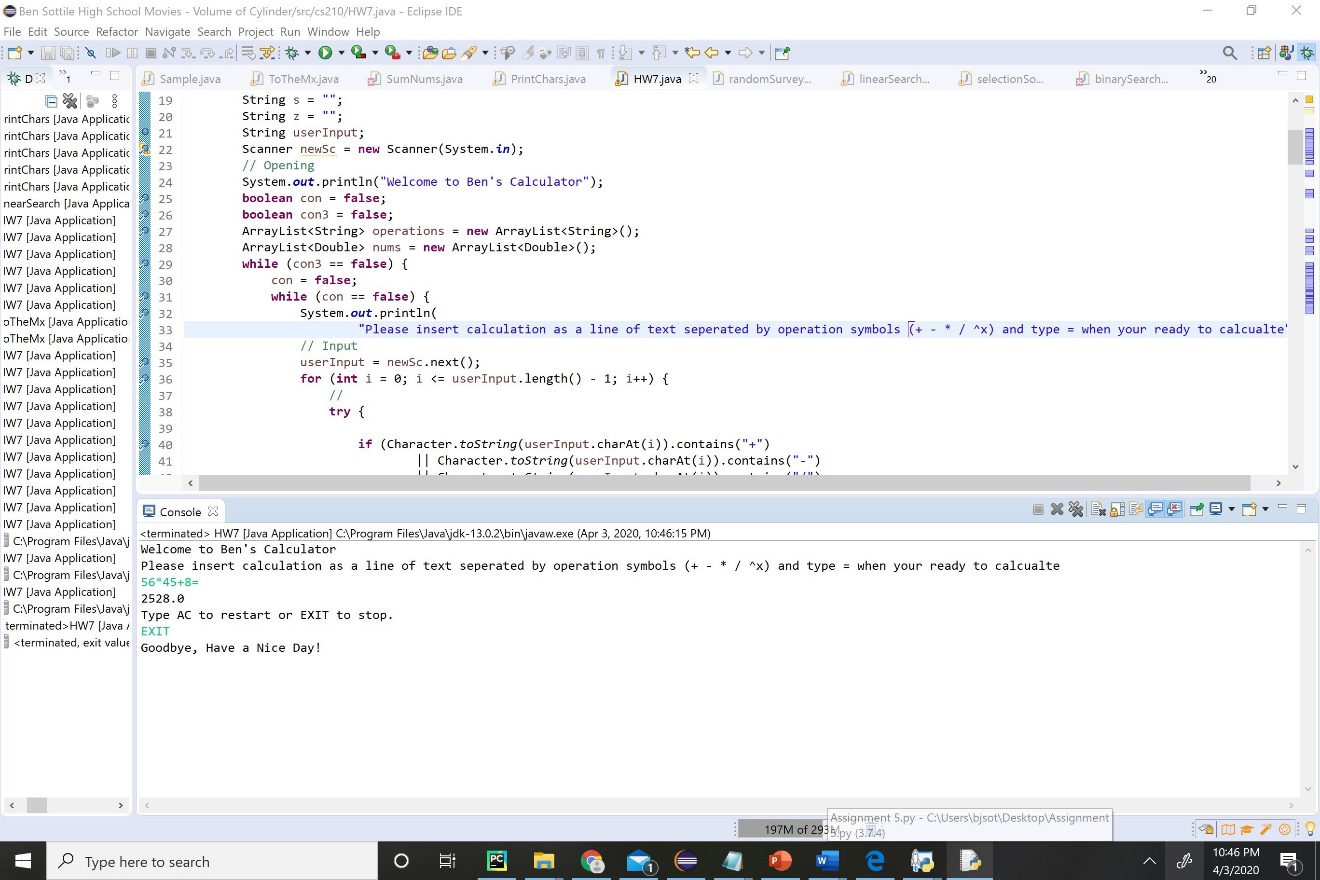
}

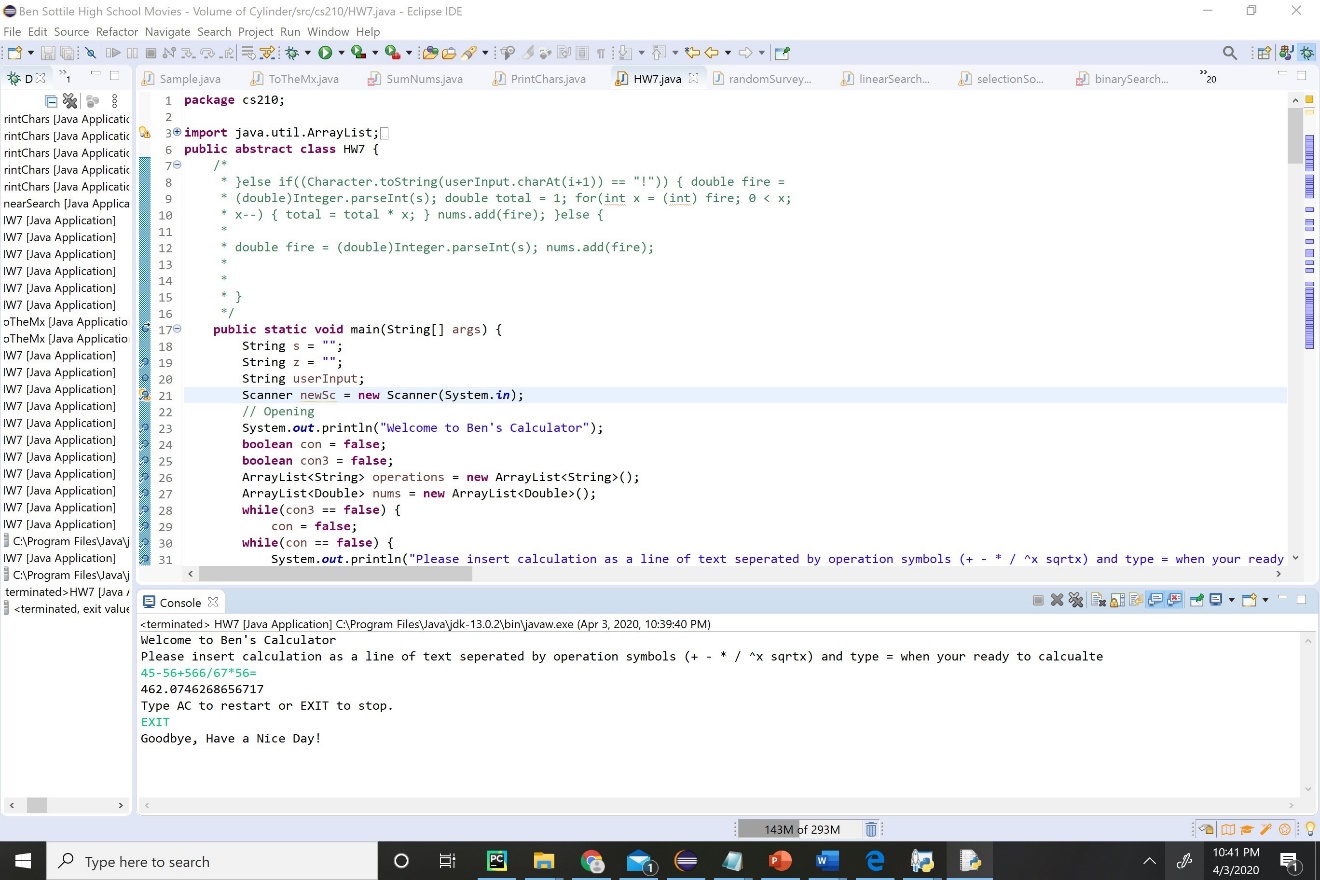
1. Testing: (10 pts)

(Describe how you test this program, and attach your results screen shots here. For this homework at least 4 test cases are required (Be clear in your results.)

I’ll I did was type random Math Problems and checking the answer with a calculator. 





5. Any additional operations like (Trig functions, Log Functions, “( )”) has 1 extra point for each operation. Please write here if you added any extra method for any extra operations.

Exponents can be calculated ^x