Lab 6 – Sentinel Loops

Part 1: Some Sentinel Loop Exercises

Find the *BlueJ* project folder named `Lab6_Start` in the `Resource` folder. Copy this folder to your home directory. Rename your copy to `Lab6_astname_firstname_Lab6`. This project contains a program which you will use for the first part of the lab.

Open the class `Part1`. The user inputs a series of positive number and echo prints the number, calculates and prints the running sum. The loop terminates when a negative number is entered and the final sum is printed. This program works as described.

1. Trace the program with some sample data (input 1, 2, 3, -1). Run the program with these inputs and compare with the results of your trace. (Compile the program if necessary).

2. Re- run the program and input -1 as the first input. What is the result? Is this what you would expect?

3. Error – Forgetting the priming read

   Edit the program by deleting the priming read. (Note: you can have the same effect if you comment out the line by placing "//" at the very beginning of the line). Compile the program. Does it compile? If not, what is the message?

4. Error – Forgetting the internal read

   Return the program to its original state and compile to make sure that it works. Now, remove the internal read and compile the program. Does it compile? If not, what is the message? If it compiled, run the program with the sample values in #1 above. What happened? Can you explain why this happened? (Note: if you get an infinite loop, terminate the program as you did in Lab 3).

Part2: Designing and Implementing a Sentinel Loop

Banks pay interest on the balance of savings accounts. For simplicity, assume that the bank pays interest on the last day of the year on the balance of the account on that day. Write a program that will input the account number and the balance of the account in a loop. It will then calculate the interest earned and print out the results according to a specified format. The loop will end if a negative account number is entered.

The (incomplete) algorithm is below:

```
INTEREST_RATE = 0.05
read ?????
while (?????)
    read openingBalance
    interest = openingBalance * INTEREST_RATE
    closingBalance = openingBalance + interest
    print (accountNo, openingBalance, interest, closingBalance)
    read ?????
```

Lab 6 – Sentinel Loops

Complete the algorithm by filling in the ?????. Write down what should replace each ?????. Once you have done so, translate the algorithm into a syntactically correct program. For this program, declare the interest rate as a named constant.

To do this, create a new class within this same project and name it `Part2`. Write out the Java code and test your program using the data as below. Do not worry about formatting the output at this time but rather ensure that the results you get are the correct results.

Next, format your output to look like the one below:

```
Interest rate = 0.05

Account  Opening      Interest     Closing
Number   Balance      Earned       Balance
1234     1000.0       50.0         1050.0
4321     200.0        10.0         210.0
5678     500.0        25.0         525.0

Thank you for using the Interest Calculator
```

Submission:

Submit the entire Java folder to the submit drive

Extra Challenge:

For those who may have more time or want an additional challenge, you can use this program as a base but add a chequing account type that has a different interest rate than the savings account. In addition to entering all the other information, the user is requested to enter the type (1 – savings, 2 – chequing). The savings and chequing interest rates are declared as named constants (values are 0.05 and 0.02, respectively) at the beginning of the program and the appropriate rate is used to calculate the interest. You may need to modify your output to include the two interest rates and add a column for the type of account.