***Lab #7: Pseudocode and Scratch Programming***

-Rebecca Skahen-

**Due: Thursday, March 27, 2014 by Midnight**

**Instructions**

* You will need your WFU-issued ThinkPad for this lab. This lab makes use of a programming language named *Scratch.* You should already have Scratch on your computer from the last lab. If not, download Scratch using the instructions under Resources -> Scratch in Sakai.
* Submit your answers to the assignment in Sakai using this document and any other material you are asked to submit in the questions, e.g., a Scratch program.

**Part 0: Background**

The purpose of this lab is to reinforce what you have learned about algorithms and pseudocode by translating them into a high-level computer language. Background comes from Chapters 6 and 7 (Computer Science Illuminated) and from lectures. For our purposes, the idea of programming is to take an algorithm in some form and translate it into *code that a computer can understand*. The computer in this case is the “compute agent” for the algorithm, much in the same way you might think of a chef as the compute agent for a recipe. We will use an excellent, introductory language, Scratch, that allows focus on problem solving and creativity rather than syntax by providing a graphical interface with drag-and-drop program components (statements). You will immediately see the ease of this approach as compared to machine or assembly code. However, you should remember that eventually your programs must be translated into machine code in order to run on your computer (not by you ☺).

When learning to program, particularly at this stage of experience, it is informative to both analyze and synthesize code. That is, read and study what others have done then write some of your own. You will be asked to describe what existing code is doing, modify code, and write your own.

**Part 1:**

For this part you will work with the following pseudocode that we discussed in class. In this part, you will be asked to follow and modify the code. Please do not delete any lines of code. Each time you modify the program, you will be adding new statements to your previous (not the original) version. If there are not enough rows in the table, just add more.

**READ inputNumber**

**SET sum to 0**

**SET maxValue to inputNumber**

**WHILE (inputNumber is not less than zero)**

**SET sum to sum + inputNumber**

**IF (inputNumber is greater than maxValue)**

**SET maxValue to inputNumber**

**READ inputNumber**

**WRITE “Max value is”, maxValue**

**Question 1.**

1.a. Identify each variable in the above code and enter its name in the left column of the below table.

1.b. Assume the following values are input to the program (10, 20, 15, 5, 55, -10). ***Note, the -10 is an end of data marker and so should not be used in any calculations.*** Complete the table by filling in the variable names and record their values as they change.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Variable Name** | **Variable Value** | | | | | | | | |
| Initial | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| Input Number | 10 | 20 | 15 |  | 5 | -10 |  |  |  |
| Sum | 0 | 10 | 30 | 45 | 50 | 105 |  |  |  |
| MaxValue | 10 |  | 20 |  |  | 55 |  |  |  |
| Output |  |  | Max | Value | Is | 55 |  |  |  |

**Question 2.**

2.a. **Without** deleting any of the existing code, modify the code to compute the sum of the values read and write it out.

READ inputNumber

SET sum to 0

SET maxValue to inputNumber

WHILE (inputNumber is not less than zero)

SET sum to sum + inputNumber

IF(inputNumber is greater than maxValue)

SET maxValue to inputNumber

READ inputNumber

WRITE “Max value is” maxValue

WRITE: “Sume of values is” , Sum

2.b. Assume the following values are input to the program (10, 20, 15, 5, 55, -10). Complete the table by filling in the variable names and record their values as they change.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Variable Name** | **Variable Value** | | | | | | | | |
| Initial | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| InputNumber | 10 | 20 | 15 |  | 5 | 55 | -10 |  |  |
| Sum | 0 | 10 | 30 | 45 | 50 | 105 |  |  |  |
| MaxValue | 10 |  | 20 |  |  | 55 |  |  |  |
| Output |  | Max | Value | Is | 55 |  |  |  |  |
|  |  | Sum | Of | Values | Is | 105 |  |  |  |

**Question 3.**

3.a. **Without** deleting any of the existing code, modify the code from the previous question (note, your **new** code above) to compute the average of the values read and write it out.

READ inputNumber

SET sum to 0

SET count to 0

SET maxValue to inputNumber

WHILE (inputNumber is not less than zero)

SET count to count +1

SET sum to sum + inputNumber

IF (inputNumber is greater than maxValue)

SET maxValue to inputNumber

READ inputNumber

SET Average to sum divided by count

WRITE “Max value is”, maxValue

WRITE: “Sum of values is”, Sum

WRITE: “The average is”,

3.b. Assume the following values are input to the program (10, 20, 15, 5, 55, -10). Complete the table by filling in the variable names and record their values as they change.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Variable Name** | **Variable Value** | | | | | | | | |
| Initial | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| InputNumber | 10 | 20 | 15 |  | 5 | 55 | -10 |  |  |
| Count | 0 | 1 | 2 | 3 | 4 | 5 |  |  |  |
| Sum | 0 | 10 | 30 | 45 | 50 | 105 |  |  |  |
| MaxValue | 10 |  | 20 |  |  | 55 |  |  |  |
| Average |  |  |  |  |  | 21 |  |  |  |
| Output |  |  | Max | Value | Is | 55 |  |  |  |
|  |  | Sum | Of | Value | Is | 105 |  |  |  |
|  |  |  | The | Average | Is | 21 |  |  |  |

**Question 4.**

4.a. **Without** deleting any of the existing code, modify the code from the previous question (note, your **new** code above) to find both the min and max values and write them out.

READ inputNumber

SET sum to 0

SET count to 0

SET maxValue to inputNumber

WHILE (inputNumber is not less than zero)

SET count to count +1

SET sum to sum + inputNumber

IF (inputNumber is greater than maxValue)

SET maxValue to inputNumber

READ inputNumber

IF (inputNumber is less than maxValue)

SET minValue to inputNumber

SET Average to sum divided by count

WRITE “Max value is”, maxValue

WRITE “Min value is”, minValue

WRITE: “Sum of values is”, Sum

WRITE: “The average is”,

4.b. Assume the following values are input to the program (10, 20, 15, 5, 55, -10). Complete the table by filling in the variable names and record their values as they change.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Variable Name** | **Variable Value** | | | | | | | | |
| Initial | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| InputNumber | 10 | 20 | 15 |  | 5 | 55 | -10 |  |  |
| Count | 0 | 1 | 2 | 3 | 4 | 5 |  |  |  |
| Sum | 0 | 10 | 30 | 45 | 50 | 105 |  |  |  |
| MaxValue | 10 |  | 20 |  |  | 55 |  |  |  |
| MinValue | 10 |  | 15 |  |  | -10 |  |  |  |
| Average |  | Max | Value | Is |  | 21 |  |  |  |
| Output |  |  | Max | Value | Is | 55 |  |  |  |
|  |  |  | Min | Value | Is | -10 |  |  |  |
|  |  | Sum | Of | Value | Is | 105 |  |  |  |
|  |  |  | The | Average | Is | 21 |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

**Part 2: Implementing Algorithms and Pseudocode in Scratch.**

For this exercise, you will implement algorithms and pseudocode in Scratch. You will be introduced to variables and selection in Scratch.

**Question 5.**

Download the program **Sum\_Two\_Nums.sb2** from Sakai. Read over the code and understand it. The program should do the following. *Start when the green flag is pressed. Ask the user to input two numbers. Sum the numbers. If the output is greater than or equal to zero, print the sum. If the sum is negative, write an appropriate message.*

* Do the following: test numbers that produce
  + a negative sum
  + a positive sum
  + a sum that equals zero

5.a. Which situation does not work (i.e., does not result in an answer)?

The situation of the “a sum that equals zero” does not work.

5.b. Where is the error in the program?

The error in the program is that there is no command if the “sum equals zero” there is only a command if the sum is greater or less than zero

5.c. Fix the error. Hint: Change the condition in the first *if* statement by replacing it with an *operator* block that has an *or* operator in it. Then test for greater than zero or equal to zero. Did you make it work?

Yes, by adding an “or” statement allowing a command if the sum is equal to zero.

5.d. Select the *Data* blocks. What are the three variables? What happens when you check and uncheck the boxes next to them?

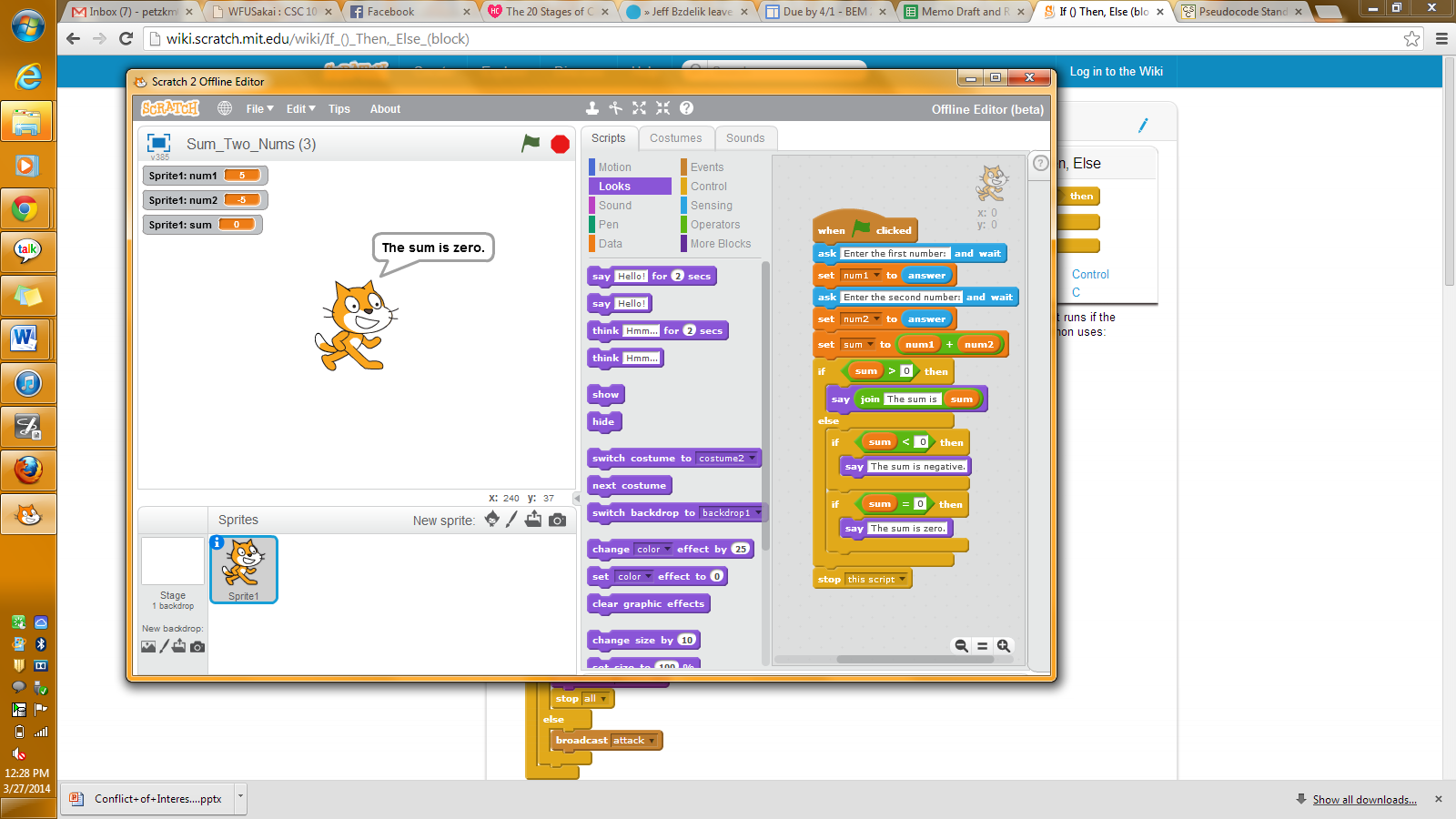
The quotation bubble of the sprite changes from the first sum to the second sum when the box is checked and unchecked.

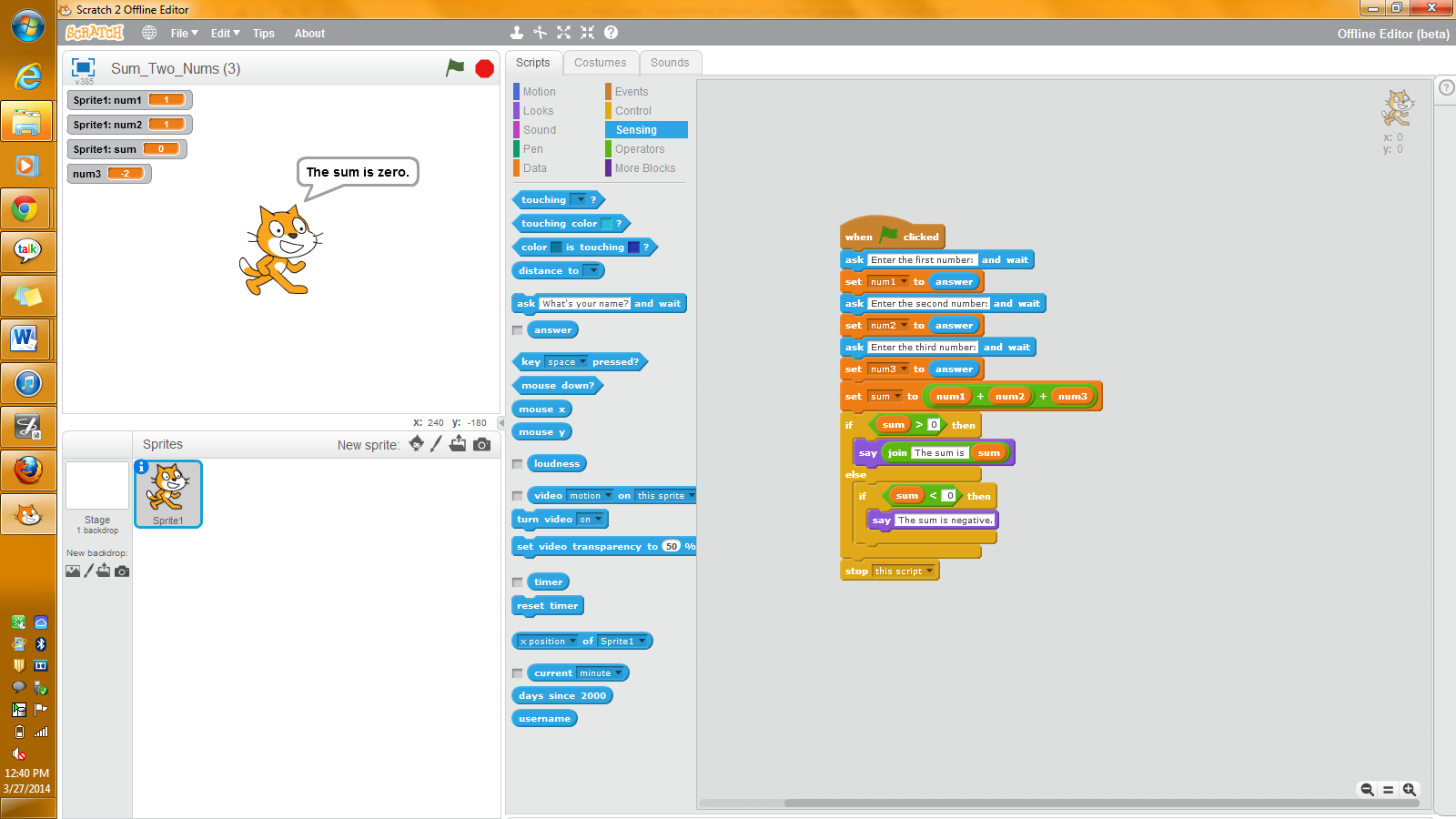
**Question 6.**

For this question, you will modify the **Sum\_Two\_Nums.sb2** Scratch program as indicated below. When complete, ***take a screenshot*** of the Scratch window and paste it where indicated. ***Be sure first to crop the image*** so as only to show the code. ***Make it big enough*** for the instructor to read.

6.a. Replace the two *if* statements with an *if-then-else* construct as discussed in class when covering pseudocode selection constructs. Paste your screenshot below. Did you make it work?

Yes it works.

6.b. Change the program to input 3 numbers and sum them and output using the same rules as above discussed at the beginning of question 5. Paste your screenshot below.



**Question 7.**

1. Write a Scratch program that implements the Pseudocode for Question 3.
2. Test the program using the inputs specified in Question 3.
3. Save the program using the name *Lab7Q7.sb2*.
4. Submit the program to Sakai.

SUMBITTED ON SAKAI