# **Lab 1: Input, Processing, and Output**

This lab accompanies Chapter 2 of *Starting Out with Programming Logic & Design*.

Name: \_\_\_\_\_\_\_\_\_\_\_\_David White\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Lab 1.1 – Algorithms**

This lab requires you to think about the steps that take place in a program by writing algorithms. Read the following program prior to completing the lab.

Critical Review

An algorithm is a set of well-designed logical steps that must take place in order to solve a problem.

The flow the algorithm takes is sequential. For example, before you process calculations, all data needed should be retrieved.

Write a program that will take in basic information from a student, including student name, degree name, number of credits taken so far, and the total number of credits required in the degree program. The program will then calculate how many credits are needed to graduate. Display should include the student name, the degree name, and credits left to graduate.

**Step 1:** Examine the following algorithm.

1. Get the student name.
2. Get the degree program name.
3. Subtract the number of credits taken so far from the required credits for the degree.
4. Get the number of credits required for the degree program.
5. Get the number of credits the student has taken so far.
6. Display the input information in Step 1 and 2.
7. Display the calculated information.

**Step 2:** What logic error do you spot and how would you fix it?

The subtraction takes place before the inputs are given. I would fix it by doing the calculation after getting the inputs.

**Step 3:** What steps require user interaction (Ex: user must type in some input)?

Step 4 and Step 5

**Lab 1.2 – Pseudocode**

Critical Review

Pseudocode is an informal language that has no syntax rules and is not meant to be compiled or executed.

The flow the program takes is sequential. For example, before you ask for input, you should display what information you want from the user.

// Comments are done by putting two forward slashes

// before the lines you want to document. Comments

// are used to explain code.

Variables are named storage locations.

Declare is the keyword used before naming a variable. Data types are: Real for decimal numbers, Integer for whole numbers, and String for a series of characters.

Follow the rules for naming variables: (1) must be one word, no spaces, (2) usually no punctuation characters, only letters and numbers, and (3) name cannot start with a number.

Display is the keyword used to print something to the screen. Any information needed to be displayed to the user should be put inside quotation marks such as:

Display "This is how you print something to the screen"

When using display to print both a string and the value of a variable, a comma is used, such as:

Display "Here is the average: ", average.

Input is the keyword used to get the user to enter data. The data value entered by the user will be placed in the variable that follows the keyword input such as *Input variableName*.

Set is the keyword used before a calculation. Standard math operators are used, such as + - \* / MOD ^. Operators can be combined in one calculation, but it is wise to group expressions together using parentheses. Remember the order of operations. Some examples are:

Set sale = price – discount

Set average = (test1 + test2 + test3) / 3

This lab requires you to think about the steps that take place in a program by writing pseudocode. Read the following program prior to completing the lab.

Write a program that will take in basic information from a student, including student name, degree name, number of credits taken so far, and the total number of credits required in the degree program. The program will then calculate how many credits are needed to graduate. Display should include the student name, the degree name, and credits left to graduate.

**Step 1**: This program is most easily solved using just five variables. Identify potential problems with the following variables declared in the pseudocode. Assume that the college has the ability to offer half credits. (Reference: Variable Names, page 39-40).

|  |  |  |
| --- | --- | --- |
| **Variable Name** | **Problem (Yes or No)** | **If Yes, what's wrong?** |
| Declare Real creditsTaken | No |  |
| Declare Real credits Degree | Yes | Variable has space |
| Declare Int creditsLeft | No |  |
| Declare Real studentName | Yes | Variable should be a string |
| Declare String degreeName | No |  |

**Step 2:** Complete the pseudocode by writing the two missing lines. (Reference: Prompting the User, page 42).

Display "Enter student name."

Input studentName

Display "Enter degree program."

Input degreeName

Display “Enter number of credits needed to graduate.”

Input creditsDegree

Display "Enter the number of credits taken so far."

Input creditsTaken

**Step 3**: What two things are wrong with the following calculation? (Reference: Variable Assignment and Calculations, page 43).

creditsLeft = creditsTaken – creditsDegree

creditsLeft is not being Set and the Order of Operations is wrong

**Step 4:**  Write the exact output you would expect from the following line of code if the user of the program enters “Bill Jones”. (Reference: Displaying Multiple Items, page 40 – 41).

Display "The student's name is ", studentName

The student’s name is Bill Jones

**Step 5:** Write the exact output you would expect from the following line of code if the user of the program enters a degree that is 63 credits in total and they have taken 40 credits. (Reference: Displaying Multiple Items, page 40 – 41).

Display "This program requires ", creditsDegree, " credits and they have taken ", creditsTaken, " so far."

**Step 6:** Complete the following pseudocode to solve the programming problem.

1. //This program takes in student information and calculates
2. //how many credits the student has left before graduation.
3. //Information is then printed to the screen.
4. //Declare variables
5. Declare Real creditsTaken
6. Declare Real creditsDegree
7. Declare Real creditsLeft
8. Declare String studentName
9. Declare String degreeName
10. //Ask for user input
11. Display "Enter student name."
12. Input studentName
13. Display “Enter name of degree.”
14. Input degreeName
15. Display “Enter number of credits for degree.”
16. Input creditsDegree
17. Display “Enter number of credits taken.”
18. Input creditsTaken
19. //Calculate remaining credits
20. Set creditsLeft = creditsDegree - creditsTaken
21. //Display student name, degree program, and credits left.
22. Display "The student's name is ", studentName
23. Display “The degree name is “, degreeName
24. Display “There are “, creditsLeft, “ credits left until graduation.”

**Lab 1.3 – Flowcharts**

Critical Review

A flowchart is a diagram that graphically depicts the steps that take place in a program. Symbols are used to depict the various steps that need to happen within a program. Flow lines are used between the symbols to indicate the flow of the program.

Ovals are used as terminal symbols, which indicate a start and stop to a program.

Parallelograms, the data symbol, are used for input and display statements.

Rectangles, the process symbol, are used for calculations and variable declarations.

On page connectors are used to link a flowchart that continues on the same page. The connecting system starts with the letter A, whereas A would appear in the two connectors that show the flow.

The statements inside the data and the process symbols can be written similarly to the statements used in pseudocode.



This lab requires you to think about the steps that take place in a program by designing a flowchart. While designing flowcharts can be done with paper and pencil, one mistake often requires a lot of erasing. Therefore, a flowcharting application such as Viso or Draw.io should be used. Read the following program prior to completing the lab.

Write a program that will take in basic information from a student, including student name, degree name, number of credits taken so far, and the total number of credits required in the degree program. The program will then calculate how many credits are needed to graduate. Display should include the student name, the degree name, and credits left to graduate.

**Step 1:** Watch the video posted in your Week 2 folder of how to create a flowchart using Visio. You may use another flowcharting tool if you’d like.

**Step 2:** Create your flowchart, and save it on your computer. You will need to upload it in Blackboard later.

**Step 3:** Insert your finished flowchart in the space below for reference later. Inside Word in the space below, select Edit and Paste.

**Lab 1.4 – Python Code**

This lab requires you to translate your work in the pseudocode and flowchart to actual code using Python. Read the following program prior to completing the lab.

Critical Review

Comments in Python are preceded by the # sign.

Input of strings into a variable is done using the *input* function. This function converts the input to a series of characters so they can be used later in the program. This is written as a statement such as:

stringVariable = input('Enter a word. ')

The *input* function always returns the user's input as a string, even if the user enters numeric data. If you want your program to read a numeric value as input, you must use a conversion function along with the *input* function. To convert input to an integer, you use the *int()* conversion function, and to convert input to a real number, you use the *float()* conversion function. For example, the following statement reads input from the user, converts that input to an integer, and assigns the integer value to a variable named number:

number = int(input('Enter a number. '))

The following statement reads input from the user, converts that input to an real number (also known as a floating-point number), and assigns the integer value to a variable named number:

number = float(input('Enter a number. '))

Equations are written similarly to the method used in pseudocode, but without the *Set* keyword. For example:

total = apples + oranges

Complex formulas should use parentheses to group processes. In addition, if input values are taken in as integers, but will be used to calculate a decimal value, they must be converted to real values. For example:

average = (test1 + test2) / 2

To display information to the screen, the *print* function is used with the string, which is written within single quotation marks. If the value of a variable needs to displayed after the string, a comma separates the two. For example:

print('The average is', average)

Write a program that will take in basic information from a student, including student name, degree name, number of credits taken so far, and the total number of credits required in the degree program. The program will then calculate how many credits are needed to graduate. Display should include the student name, the degree name, and credits left to graduate.

**Step** **1:** Examine the following line of code. What do you expect as output to the screen?

studentName = input('Enter student name. ')

Enter student name.

**Step 2:** Examine the following line of code. What type of value do you expect the user of the program to enter?

creditsDegree = int(input('Enter credits required for degree.'))

number

**Step 3:** If the user of the program types Bill Jones to the question in Step 1, what do you expect the output to the screen to be when the following line of code processes?

print('The student\'s name is', studentName)

The student’s name is Bill Jones

**Step 4:** Examine the following code. If the program requires 63 credits, and the student has 20 left, what do you expect the output to the screen to be?

print('The program requires', creditsDegree,

'credits and they have taken', creditsTaken,

'credits so far.')

The program requies 60 credits and they have taken 43 credits so far.

**Step 5:** Start the IDLE Environment for Python. If the Edit window for entering code does not come up, go to Options, Configure IDLE, click on the General tab, and under Startup Preferences select Open Edit Window. Close and reopen the Environment. Prior to entering code, save your file by clicking on File and then Save. Select your location and save this file as *Lab1-4.py*. Be sure to include the .py extension.

**Step 6:** Code should start with documentation. A sample program header is provided in your Week 2 folder. You must use it going forward in ALL of your programs. Document the first few lines of your program to include your name, the date, and a brief description of what the program does. Each line that you want to comment out must begin with a # sign. For example:

# Sally Smith

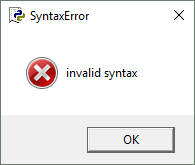
# January 15

# This program ...

**Step 7:** After documentation, enter the following line of code into your program.

studentName = input('Enter student name. ')

**Step 8:** On the menu, select Run and then Run Module. Observe your program in action. If you get a syntax error, **you must fix it** before you are able to run your program. Click OK and review the highlighted syntax error to fix it.



**Step 9:** Repeat Step 7, but change the statement so that it asks the user to enter their degree name. It is up to you whether you want to repeat Step 8 each time you code a line. It is recommended for beginning programmers so they can immediately identify syntax errors. Also, one syntax error at a time seems better than many all at once.

**Step 10:** Next, you should write the code that will ask the user how many credits are required in the degree. This can be done using the *input* function since it is a numeric value. Enter the following line of code into your program.

creditsDegree = int(input('Enter the number of credits required for the degree. '))

**Step 11:** Repeat Step 10 but change the statement so that it asks the user to enter the number of credits they have taken so far.

**Step 12:** Next, add your calculation. This is done very simply with the following code.

creditsLeft = creditsDegree – creditsTaken

**Step 13:** Add the following line of code to your program.

print('The student's name is', studentName)

**Step 14:** If you have not tested your program in a while, now is a good time to try it out. Go to Run and Run Module and observe what happens. ***SYNTAX ERROR!***

**Step 15:** While nothing stands out as being wrong in Step 14, notice that the word student's is actually causing the problem. To the language, the apostrophe looks as if it is the end of the statement. Since it is not, it must be quoted out by putting a \ in front of it. Change the line to the following.

print('The student\'s name is', studentName)

**Step 16:** Finish your code by printing the remaining of the requested statements. Your final output might look like the following.

Enter student name. Bill Jones

Enter degree name. Computer Programming

Enter the number of credits required for the degree. 63

Enter the number of credits taken so far. 24

The student's name is Bill Jones

The degree name is Computer Programming

There are 39.0 credits left until graduation.

**Step 17:** When your code is complete and runs properly, on the Menu, go to Edit and then Select All, then Edit and Copy. Paste the code below. Save a copy of your .py file for upload to Blackboard.

#######################################################

# Name: David White

# Class: CIS-1400

# Assignment: Lab 1-4

# File: lab1-4.py

# Purpose: Average 3 test scores

#######################################################

print('\n\*\* David White \*\*\n') # Display author's name

#Declare variables

StudentName = str(input('Enter student name. '))

DegreeName = str(input('Enter Degree name. '))

CreditsDegree = int(input('Enter the number of credits required for the degree. '))

CreditsTaken = int(input('Enter the number of credits taken. '))

#Process

CreditsLeft = CreditsDegree - CreditsTaken

#output

print('The student\'s name is ', StudentName)

print('The Degree name is ', DegreeName)

print('There are', CreditsLeft ,'credits left until graduation')

**Lab 1.5 – Programming Challenge 1 – Team Average**

Write the Algorithm, Pseudocode, Flowchart, and Python code for the following programming problem.

**Team Average**

A college wants you to write a program for them that will calculate the average number of wins for their football team over the past five years. The user of the program should be able to enter the number of wins each year. The program will calculate the average number of wins during that five year period and display that information to the screen.

**The Algorithm**

1. **Get number of wins from year 1**
2. **Get number of wins from year 2**
3. **Get number of wins from year 3**
4. **Get number of wins from year 4**
5. **Get number of wins from year 5**
6. **Average the number of wins from each year**
7. **Display win average**

**The Pseudocode**

//This program calculates the average number of wins during a 5

//year period and displays that information on the screen

//declare variables

Declare Int wyear1

Declare Int wyear2

Declare Int wyear3

Declare Int wyear4

Declare Int wyear5

Declare Real avgwin

//Ask for user input

Display “Enter number of wins from year 1”

Input wyear1

Display “Enter number of wins from year 2”

Input wyear2

Display “Enter number of wins from year 3”

Input wyear3

Display “enter number of wins from year 4”

Input wyear4

Display “Enter number of wins from year 5”

Input wyear5

//calculate average

Set avgwin = (wyear1 + wyear2 + wyear3 + wyear4 + wyear5) / 5

//Display average

Display “You have won an average of “, avgwin, “ games per year.”

**The Flowchart**



**The Python Code**

#######################################################

# Name: David White

# Class: CIS-1400

# Assignment: Lab 1-5

# File: lab1-5.py

# Purpose: Calculate the avergage number of wins during a 5 year period

#######################################################

print('\n\*\* David White \*\*\n') # Display author's name

#Declare Variables

WYear1 = int(input('Enter number of wins from year 1. '))

WYear2 = int(input('Enter number of wins from year 2. '))

WYear3 = int(input('Enter number of wins from year 3. '))

WYear4 = int(input('Enter number of wins from year 4. '))

WYear5 = float(input('Enter number of wins from year 5. '))

#calculate average

AvgWin = (WYear1 + WYear2 + WYear3 + WYear4 + WYear5) / 5

#output

print('You have won an average of', AvgWin, 'games per year.')