# **Lab 6: Repetition Structures**

This lab accompanies Chapter 5 of *Starting Out with Programming Logic & Design*.

Name: David White

**Lab 6.1 – For Loop and Pseudocode**

|  |
| --- |
| Critical Review  A count-controlled loop iterates a specific number of times. Although you can write this with a While or a Do-While loop as performed in Lab 5, most programming languages provide a loop known as the For loop. This loop is specifically designed as a count-controlled loop.  The process of the For loop is:   * The loop keeps a count of the number of times that it iterates, and when the count reaches a specified amount, the loop stops. * A count-controlled loop uses a variable known as a counter variable to store the number of iterations that it has performed. * Using the counter, the following three actions take place (Initialization, Test, and Increment).   The pseudocode for a for statement looks as follows:  For counterVariable = startingValue to maxValue  Statement  Statement  Statement  Etc.  End For |

This lab requires you to implement a count-controlled loop using a For statement.

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

Constant Integer MAX\_HOURS = 24

Declare Integer hours

For hours = 1 to MAX\_HOURS

Display "The hour is ", hours

End For

**Step 2:** Explain what you think will be displayed to the screen in Step 1. (Reference: For Statement, page 239):

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_The current hour\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Step 3:** Write a For loop that will print 60 minutes to the screen. Complete the missing lines of code.

Constant Integer MAX\_MINUTES = 60

Declare Integer minutes

For minutes = 1 to MAX\_MINUTES

Display “The minute is “, minutes

End For

**Step 4:** Write a For loop that will print 60 seconds to the screen. Complete the missing lines of code.

Constant Integer MAX\_SECONDS = 60

Declare Integer seconds

For seconds = 1 to MAX\_SECONDS

Display “The current second is “, seconds

End For

**Step 5:** For loops can also be used to increment by more than one. Examine the following code.

Constant Integer MAX\_VALUE = 10

Declare Integer counter

For counter = 0 to MAX\_VALUE Step 2

Display "The number is ", counter

End For

**Step 6:** Explain what you think will be displayed to the screen in Step 5. (Reference: Incrementing by Values Other than 1, page 244):

The even numbers between 1 and 10

**Step 7:** Write a For loop that will display the numbers starting at 20, then 40, then 60, and continuing the sequence all the way to 200.

Constant Integer MAX\_VALUE = 200

Declare Integer counter

For counter = 0 to MAX\_VALUE Step 20

Display "The number is ", counter

End For

**Step 8:** For loops can also be used when the user controls the number of iterations. Examine the following code:

Declare Integer numStudents

Declare Integer counter

Display "Enter the number of students in class"

Input numStudents

For counter = 1 to numStudents

Display "Student #", counter

End For

**Step 9:** Explain what you think will be displayed to the screen in Step 8. (Reference: Letting the User Control the Number of Iterations, page 248):

a count of the number of students

**Step 10:** For loops are also commonly used to calculate a running total. Examine the following code.

Declare Integer counter

Declare Integer total = 0

Declare Integer number

For counter = 1 to 5

Display "Enter a number: "

Input number

Set total = total + number

End For

Display "The total is: ", total

**Step 11:** Explain what you think will be displayed to the screen in Step 10. (Reference: Calculating a Running Total, page 255):

The sum of numbers 1 through 5

**Step 12:** Write the missing lines for a program that will allow the user to enter how many ages they want to enter and then find the average.

Declare Integer counter

Declare Integer totalAge = 0

Declare Real averageAge = 0

Declare Integer age

Declare Integer number

Display "How many ages do you want to enter: "

Input number

For counter = 1 to number

Display "Enter age: "

Input age

Set totalAge = age + totalAge

End For

averageAge = totalAge / number

Display "The average age is ", averageAge

**Lab 6.2 –For Loop and Flowcharts**

This lab requires you to convert various pseudocode steps in Lab 6.1 to three separate flowcharts. Use Visio or Draw.io to create your diagrams.

**The Seconds Counter**

**Step 1:** Name this flowchart *Lab 6-2Seconds*.

**Step 2:** The first loop to code is the pseudocode from Step 4, Lab 6.1. This loop will print 60 seconds to the screen. The complete pseudocode is below:

Constant Integer MAX\_SECONDS = 60

Declare Integer seconds

For seconds = 1 to 60

Display "The second is ", seconds

End For



**The Accumulator**

**Step 1:** Name this flowchart *Lab 6-2Accumulator*.

**Step 2:** The next loop to code is the pseudocode from Step 10, Lab 6.1. This loop will take in a number and accumulate the total. The complete pseudocode is below:

Declare Integer counter

Declare Integer total = 0

Declare Integer number

For counter = 1 to 5

Display "Enter a number: "

Input number

Set total = total + number

End For

Display "The total is total: ", total



**The Average Age**

**Step 1:** Name this flowchart *Lab 6-2AverageAge*.

**Step 2:** The next loop to code is the pseudocode from Step 12, Lab 6.1. This loop will take in various amounts of ages and then find the average. The complete pseudocode is below:

Declare Integer counter

Declare Integer totalAge = 0

Declare Real averageAge = 0

Declare Integer age

Declare Integer number

Display "How many ages do you want to enter: "

Input number

For counter = 1 to number

Display "Enter age: "

Input age

Set totalAge = totalAge + age

End For

Set averageAge = totalAge / number

Display "The average age is ", averageAge



**Lab 6.3 – Python Code**

Critical Review

You use the for statement to write a count-controlled loop. In Python, the for statement is designed to work with a sequence of data items. When the statement executes, it iterates once for each item in the sequence. The general format is as follows:

for *variable* in [*value1*, *value2*, *etc.*]:

*statement*

*statement*

*etc.*

Using the range function

When it is too cumbersome to print all the values to be displayed, Python has a range function that can be used. If you pass one argument to the range function, that argument is used as the ending limit of the list. If you pass two arguments to the range function, the first argument is used as the starting value of the list and the second argument is used as the ending limit. Here are two examples:

|  |  |
| --- | --- |
| for num in range(5):  print num  This code will display the following:    0  1  2  3  4 | for num in range(1, 5):  print num  This code will display the following:  1  2  3  4 |

Letting the User Control the Number of Iterations

Sometimes the programmer needs to let the user control the number of times that a loop iterates. This is done by first letting the user enter how many times they want their loop to execute. Then, the range function is used to control the iterations. It is important to use the starting value of 0 for the loop to execute the exact number of times. The general format is as follows:

number = int(input('How many iterations do you want: '))

for counter in range(0, number):

*Statements…*

*Statements…*

The goal of this lab is to convert all flowcharts in Lab 6.2 to Python code.

**Step 1:** Start the IDLE Environment for Python. Prior to entering code, save your file by clicking on File and then Save. Select your location and save this file as *Lab6-3.py*. Be sure to include the .py extension.

**Step 2:** Document the first few lines of your program to include your name, the date, and a brief description of what the program does.

**Step** **3:** Start your program with the following code for main:

# Lab 6-3 Practicing for loops

# the main function

def main():

# A Basic For loop

# The Second Counter code

# The Accumulator code

# The Average Age code

# calls main

main()

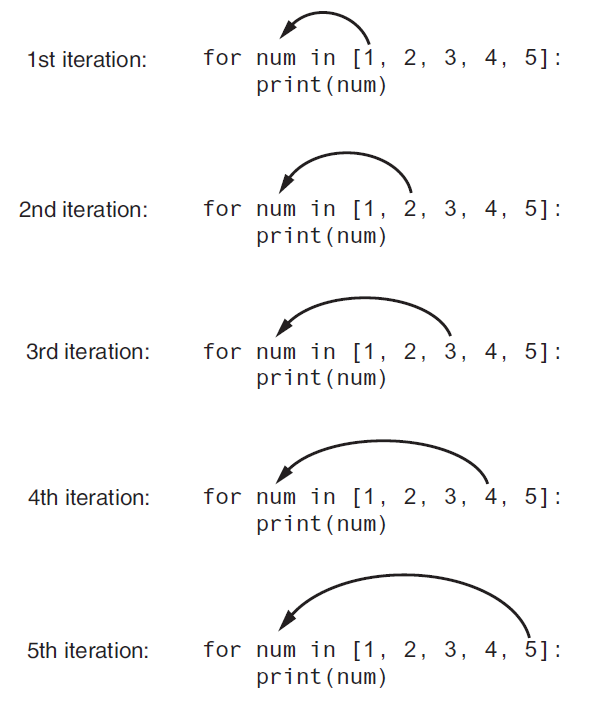
**Step 4:** Under the documentation for A Basic For Loop, add the following lines of code:

print('I will display the numbers 1 through 5.')

for num in [1, 2, 3, 4, 5]:

print num

On the first iteration, 1 is placed into the variable num and num is then printed to the screen. The process is continued as follows:



Execute your program. Notice that the output is as follows:

>>>

I will display the numbers 1 through 5.

1

2

3

4

5

>>>

**Step 5:** The next loop to code is the Second Counter code. This loop can be processed in the same way as Step 4; however, it would take a long time to write 1 through 60 in the for loop definition. Therefore, the range function should be used to simplify the process. Write a for loop that has a range from 1 to 61. If you stop at 60, only 59 seconds will be printed. If you only provide one argument, the starting value will be 0. (Reference the Critical Review section above for the exact syntax.)

**Step 6:** The next loop to code is the Accumulator code. Start by initializing a total variable to 0. This must be done in order to accumulate values.

**Step 7:** The next step is to write a for loop that iterates 5 times. The easiest way to do this is the following.

for counter in range(5):

**Step 8:** Inside the for loop, allow the user to enter a number. Then, add an accumulation statement that adds the number to total. In Python, the range function determines the number of iterations, so it is not necessary to manually increment counter.

**Step 9:** Outside of the for loop, use a print statement that will display the total.

**Step 10:** Compare your sample input and output to the following:

Enter a number: 54

Enter a number: 32

Enter a number: 231

Enter a number: 23

Enter a number: 87

The total is 427

**Step 11:** The final loop to code is the Average Age code. Start by initializing totalAge and averageAge to 0. (Reference the Critical Review section above on Letting the User Control the Number of Iterations).

**Step 12:** The next step is to ask how many ages they want to enter. Store the answer in the number variable.

**Step 13:** Write the definition for the for loop using the range function such as:

for counter in range(0, number):

**Step 14:** Inside the for loop, allow the user to enter an age.

**Step 15:** Inside the for loop, add the code that will accumulate age into the totalAge variable.

**Step 16:** Outside of the loop, calculate the average age as averageAge = totalAge / number.

**Step 17:** Outside of the loop, display the averageAge variable to the screen.

**Step 18:** Compare your sample input and output to the following:

How many ages do you want to enter: 6

Enter an age: 13

Enter an age: 43

Enter an age: 25

Enter an age: 34

Enter an age: 28

Enter an age: 43

The average age is 31

>>>

**Step** **18:** Execute your program so that all loops work and paste the final code below

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

# Name: David White

# Class: CIS-1400

# Assignment: Lab 6-3

# File: lab6-3.py

# Purpose: practice for loops

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

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

# main function

def main():

# basic for loop

print('I will display the number 1 through 5')

for num in [1, 2, 3, 4, 5]:

print(num)

# end for

print()

# second counter code

for seconds in range(1, 61):

print('The second is', seconds)

# end for

print()

# accumulator

total = int(0)

for counter in range(5):

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

total = total + number

# end for

print('The total is:', total)

print()

# average age

totalAge = int(0)

numbers = int(input('How many ages do you want to enter: '))

for counter in range(0, numbers):

age = int(input('Enter an age: '))

totalAge = totalAge + age

averageAge = int(totalAge / numbers)

print('the average age is:', averageAge)

# call main

main()

**Professor Rakestraw’s Calculator – 30 points**

**Directions:** Create a flowchart for the problem below. Once your flowchart is complete, create Python code *following the design in your flowchart*. There are many different ways to design this solution!

**Problem:**  
In my CIS 1150 class, I give lots of quizzes! Often, students come by to ask me what their average score is. My problem is that the number of quizzes and tests my students take changes on a daily basis, so I need a flexible program that students can use themselves, or I can use to compute averages for multiple students at once.

**What I Need:**  
Write a program that will allow a student or a teacher to calculate the average test score for a certain number of students. The user of the program can enter the number of students who took the test, and then the score for each student. Your program will then calculate the average score and print out the results. Your program must use the appropriate loop, modules, and run multiple times for different sets of test scores.

Your sample output might look as follows:

How many students took the test: 11

Enter their score: 22

Enter their score: 75

Enter their score: 93

Enter their score: 75

Enter their score: 91

Enter their score: 84

Enter their score: 87

Enter their score: 23

Enter their score: 57

Enter their score: 61

Enter their score: 98

The average test score is 70

Do you want to end program? (Enter no to process a new set of scores): yes

You must use the pseudocode skeleton I provided below for main as your starting point. Please note that the “?” indicate that something is missing!

**The Pseudocode**

Module main()

// Declare and initialize all local variables

Declare string endProgram = n

Declare integer totalScores = 0

Declare Real averageScores = 0

Declare integer score = 0

Declare integer number = 0

Declare integer counter = 1

// Loop to run program again

While endProgram == n

//reset variables

totalScores = 0  
averageScores = 0

counter = 1

score = 0

//calls functions

Call getNumber(number)

Call getScores(totalScores, number, score, counter)

Call getAverage(totalScores, number, averageScores)

Call printAverage(averageScores)

Display “Do you want to end the program? (y/n)”

Input endProgram

End While

End Module

Module getNumber(Integer Ref number)

Display “How many students took the test: “

Input number

End Module

Module getScores(Real Ref totalScores, Integer number, Real score,

Integer counter)

For counter <= number

Display “enter their score: “

Input score

Set totalScores = totalScores + score

End For

End Module

Module getAverage(Real totalScores, Integer number,

Real Ref averageScores)

Set averageScores = totalScores / number

End Module

Module printAverage(Real averageScores)

Display “The average test score is “, averageScores

End Module