|  |  |
| --- | --- |
| **Activity No. 3** | |
| **Control Structures** | |
| **Course Code:** CPE009 | **Program:** BSCPE |
| **Course Title:** Object-Oriented Programming | **Date Performed:** 11/21/2019 |
| **Section:** CPE12FB2 | **Date Submitted:** 11/21/2019 |
| **Name:** Julongbayan, Dan Angelo A. | **Instructor:** Sir Razon |
| **1. Objective(s):** | |
| This activity aims to familiarize students in implementing control structures in Python | |
| **2. Intended Learning Outcomes (ILOs):** | |
| The students should be able to:   * 1. Create a Python program that can change its output based on different conditions   2. Use the different iterative statements in a Python program | |
| **3. Discussion:** | |
| Control structures allows programmers to change the flow or sequence of the code depending on specific conditions that may come from user input or from other data or variables within the program in runtime execution.  **Conditional Statements** allow the flow of a program to change if a certain condition was met. Conditional statements need to evaluate Boolean Expressions which returns a True or False value to determine if a condition was met or not.  A **Boolean Expression** can be constructed by using the following Relational and Boolean operators:  Relational Operator Name  == Equals  > Greater than  < Less than  >= Greater than or Equal to  <= Less than or Equal to  != Not equal to  Boolean Operator Name  and Logical AND Operator  or Logical OR Operator  not Logical NOT Operator  **Relational operators** are commonly used to compare numerical values other than == which can also be used to compare Strings. In some cases, multiple Boolean Expressions may need be combined using **Boolean Operators** to create programs that can handle more complex logic such as compounding multiple Boolean expressions.  There are three keywords used to implement conditional statements in Python which are: **if**, **elif**, and **else** keywords. Conditional statements in Python use indentations to indicate which codes belong under a given condition. Given a conditional statement that only needs to handle one condition a one-way **if** statement can be used following the syntax:    To handle two or more conditions, the **elif** statement can be used under the if statement to indicate alternate conditions that will execute on the condition that the Boolean expression of the if statement is false. | |

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

|  |
| --- |
| The following shows the syntax of an if elif statement to handle two specific conditions:    Multiple elif statements can also be added.    For scenarios where the program will need to handle conditions not specific to the Boolean Expressions written in the program. An else statement can be added to the if statement or if elif statement. The else statement can act as a default when all conditions specified were false. The syntax for an if else statement is shown:    To include the else keyword in an if elif statement the syntax is as follows:    Iterative Statements allows the flow of a program to repeat certain blocks of code and even the entire program based on certain conditions. Iterative statements are implemented in two ways in Python.  **While Statements**  The structure of Python’s while loop statement is similar to that of other languages. Here is the syntax:    The while loop will proceed and repeat the code indented under it while the Boolean expression is True. The While loop will terminate or stop repeating once the Boolean Expression is false. Values in a While loop should change so as it only repeats  for a finite amount of iterations. While loops whose Boolean Expressions will never be false is called an Infinite Loop which is |

|  |
| --- |
| will repeatedly loop the code inside it until the program is externally closed. An infinite loop is considered to be a logical error in a program.  **For Statements**  Python includes a for loop statement for more concise iteration over a sequence of values. The syntax of this statement is:    A For Loop in Python unlike in other programming languages loops through a sequence or iterable object. The iterable object can be a String, List, Tuple, Dictionary, and a Generator such as the range() function. The range() function is a common generator function used with Python’s For Loop which generates numbers from 0 up to the specified value minus 1.  An example of its syntax and use is shown below:      This example code will generate numbers 0 to 9 because by default the range function starts from 0 and ends at n – 1. These codes will be seen and tried in more detail in the laboratory activity. |
| **4. Materials and Equipment:** |
| Desktop Computer with Anaconda Python Windows Operating System |
| **5. Procedure:** |
| **Relational and Boolean Operators**   1. Use two variables a and b and assign the following values to the two variables then display the output at each set of values as specified in the table below. 2. Use three variables a=1, b=2 and c=3 and then run the following print() functions below.    1. print(a<b and c==3)    2. print(a+b==2 and c<=3)    3. print(a<b or c==3)    4. print(a+b==2 or c<=3) 3. Write your observations per condition and experiment with the operators by changing the values of a, b, and/or c. |

|  |  |
| --- | --- |
| Values of a and b | Codes to display per set |
| 1, 2 | print(a == b) print(a > b) print(a >= b) print(a < b) print(a <= b) print(a != b) |
| 5, 0 |
| -1, -1 |
| 2, 2.0 |
| “same”, ‘same’ |
| (1,2,3), (1,2,4) |

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |
|  |
|  |
|  |
|  |

|  |
| --- |
| **Conditional statements**  Suppose that we have a sensor that reads the temperature inside a large machine. We want to create a program that will print a message on the control screen to inform the operator of the status of the machine.  **Scenario 1:** If the temperature is greater than 100**°**C display the status code 10 and the message  “Too hot – turn off the machine”   1. Declare a variable named **statusCode** and assign it an initial value of 0 2. Declare a variable named **machineTemp** and assign it an initial value of 0.0 3. Receive a temperature input from the user and assign the value to the variable **machineTemp.**   Note: The received value must be converted to float.   1. Type the code displayed below:     Note: Observe the different way of displaying the value unlike the last activity. This is known as an f-string which is a new way to do string formatting for Python 3.7 and above.   1. Complete and run the program. Remember to note your observations.   **Scenario 2:**   * 1. If the temperature is greater than 100**°**C display the status code 10 and the message   “Too hot – turn off the machine”   * 1. If the temperature is not greater than 100**°**C display the status code 2 and the message   “Normal reading – No action required”   1. Modify the code snippet from step#4 and add a new condition that handles scenario 2b. Hint: This scenario acts as a default condition when scenario a is not true. 2. Complete and run the program. Remember to note your observations.   **Scenario 3:**   * 1. If the temperature is greater than 100**°**C, display the status code 10 and the message   “Too hot – turn off the machine”   * 1. If the temperature is not greater than 100**°**C, display the status code 2 and the message   “Normal reading – No action required”   * 1. If the temperature is in the range of 90**°**C and 100**°**C, display the status code 5 and the message   “Critical temperature – Monitor the machine, do not leave the area”   1. Modify the code snippet from step#4 and #6 and add a new condition that handles scenario 3c. Hint: This will require compounded or join multiple Boolean expressions. 2. Complete and run the program. Remember to note your observations.   **Iterative statements**  Suppose that you are developing a program for a security door that implements a lock using an RFID card system. A Card will be tapped and scanned at the door and only registered cards will be able to unlock the door. We want to create a program that will display a message on the small monitor placed above the RFID scanner then unlock the door based on the data read from the RFID card. |

|  |
| --- |
| **Scenario 1:** The program should only continuously accept and read cards. As of now there is only one registered card. The program should not terminate unless a special card with the code “sh00” is read by the system.   1. Declare two variables named **scannedCode,** and **registeredCode** and assign both an initial value of “” 2. The program should ask for the **registeredCode** only once in the program.   Note: The **registeredCode** can be any string value (combinations of numbers and letters).   1. Type the following code below:      1. Complete and run the program. Remember to note your observations.   **Scenario 2:** The program should only repeatedly accept invalid cards up to three times. Afterwards it should shutdown automatically and be restarted by Authorized personnel. As of now there is only one registered card. The program should still terminate using special card with the code “sh00”.   1. You may copy or modify the previous code block. Declare a new variable named **counter** and assign a value of 3. 2. The program should still ask for the **registeredCode** only once in the program.   Note: The **registeredCode** can be any string value (combinations of numbers and letters).   1. Modify the code in step#3 so that the code may look similar to the one below:      1. Complete and run the program. Remember to note your observations.   **Scenario 3:** The program should only repeatedly accept invalid cards up to three times. Afterwards it should shutdown automatically and be restarted by Authorized personnel. As of now there are three registered cards. The program does not ask anymore the registered codes but instead the codes are initialized in the program. The program should still terminate using special card with the code “sh00”.  1. You may copy or modify the previous code block. Change the name of registeredCode to registeredCodes and assign the values as follows:    You may modify the code1, code2, code3 with values that you want. |

|  |
| --- |
| 1. Modify the code in the previous steps to look similar to the image below     Note: Observe the change in comparison of the if statement.   1. Complete and run the program. Remember to note your observations. |
| **6. Supplementary Activity:** |
| **Tasks**   1. **Write a program that would construct a table of values (first 10 values from the origin) using a for loop and range() function for the following Mathematical functions:**    1. Parabola:    2. Square root function:    3. Absolute-value function:   Note: for c. 5 values should come from the negative side and 5 values should come from the positive side (Ex. -5 -4 -3 -2 -1 and 1 2 3 4 5). Using/importing a library is not allowed.  Recall a table of values:  x y  -2 2  -1 1  0 0  1 1  2 2  You may refer to the documentation about the range function here: <https://docs.python.org/3/tutorial/controlflow.html>   1. **Write a program that determines the mean or average value of a given set of numbers. Hint:** Do not use any function that was not discussed or given in the laboratory activity. |
| **7. Conclusion:** |
| This lab activity has showed me just how powerful conditional statements are paired with relational or boolean operators and especially if its under a loop. It has also taught me additional syntax that can be useful in later activities specifically the use of f before the quotation marks for the print function to enable the output of variables as long as it is enclosed in curly braces. |
| **8. Assessment Rubric:** |

Screenshots:



































