HOLMES INSTITUTE FACULTY OF HIGHER EDUCATION	 INSPIRE ACHIEVE ENGAGE	HOLMES INSTITUTE Cum Propris Suis Alis Volat

Assessment Details and Submission Guidelines	
Trimester	T1 2019
Unit Code	HS1031
Unit Title	Introduction to Programming
Assessment Type	Group Assignment
Assessment Title	Assignment II
Purpose of the assessment (with ULO Mapping)	Assess student's ability to analyse programming problems and design and implement solutions as part of a developer team.
Weight	15 % of the total assessments
Total Marks	15
Word limit	N/A
Due Date	Week 10
Submission Guidelines	• There are five questions in this assignment which require you to write and submit five Python scripts. Similar to what you did in assignment I, please save each script in two different formats: **.py** and **.txt** prior to submission. For example, for the first question you need to submit **unique.py** and **unique.txt**. • Code must be appropriately commented. Make sure to add comments at each segment of your code to explain what it does. • You are allowed to use built-in library functions or methods if required. • Make sure that your code runs successfully for all possible entries. • Try to approach the solution with the least number of steps. Your code must be clear, logical, and easy to understand. • All work must be submitted to Blackboard by the due date (**Tuesday, 28 May 2019 11:55PM**). • You are encouraged to avoid last minute submission so that you do not run into technical problems. • You can only submit once. Thus, ensure that your work is final prior to submission. • This is a group project. Please make sure to divide tasks equally among the group. Group members need to collaborate to make sure that all programs run successfully and are appropriately commented.

Assignment II Specifications

Purpose:

This assignment evaluates your understanding of basic programming principles using Python programming language. In particular, this assignment assesses your ability to develop algorithms to solve simple problems, successfully develop and run python programs, and your ability to write meaningful comments when required. The assignment also provides a platform for students to work together in groups to develop solutions, which resembles how complex programming problems are solved in real-life.

Assignment Structure should be as the following:

Marking criteria

Marking criteria	Weighting
Appropriate commenting	5%
Sound logic	5%
Code running successfully	5%
TOTAL Weight	**15%**
Assessment Feedback:	

1. Write a program named **unique.py**. The program takes as an input a **text file**. Your program should print all the unique words in the file in alphabetical order.

2. Write a program named **frequency.py.** The program takes as an input a **text file**. Your program should print all the unique words in the file and their frequencies in alphabetical order.

3. Define a function **conversion.py** which takes a decimal number as an input, and returns its representation in a given base as a string (i.e. binary, octal, or hexadecimal string). The function uses a look-up table (dictionary) to map decimal integers to digits in the targeted base. For your convenience, the look-up table is provided below. A main function with test statements are also provided. Hint: figure out how base 10 can be converted to other bases, then write the algorithm.

   ```
   lookUp = {0 : '0', 1 : '1', 2 : '2', 3 : '3', 4 :'4', 5 : '5', 6 : '6', 7 : '7', 8 : '8', 9 : '9', 10 : 'A', 11 : 'B', 12 : 'C', 13 : 'D', 14 : 'E', 15 : 'F'}

   def main():
       """Tests the function."""
       print(decimalToRep(10, 10))
       print(decimalToRep(10, 8))
       print(decimalToRep(10, 2))
       print(decimalToRep(10, 16))

   # The entry point for program execution
   if __name__ == "__main__":
       main()
   ```

4. A list is said to be sorted ascendingly if it is either empty or if each item (except the last one) is less than or equal to the item prior to it. Define a function **sorted.py** which takes a list as an input and returns **True** if the list is sorted and **False** otherwise.

5. Define a function **myRange.py**, which behaves like the standard range function in python and takes both required and optional arguments. Do **not** use the standard Python's range function in your solution. You need to develop your own range function. Read Python's help on the standard range function to understand its arguments. Assign **None** as a default value for the two optional arguments. If the two optional arguments equal **None,** that means the range function has been called with just the **stop** value. If only the third argument equals **None**, this means the function has been called with both, the **start** and **stop** values. Therefore, the first part of your code determines what the parameters are while the rest of the code uses these parameters to build-up a list by counting up or down.