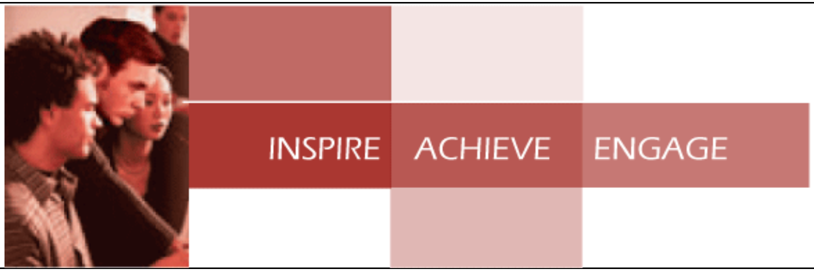
HOLMES INSTITUTE FACULTY OF HIGHER EDUCATION	

Assessment Details and Submission Guidelines	
Trimester	T1 2019
Unit Code	HS1031
Unit Title	Introduction to Programming
Assessment Type	Individual Assignment
Assessment Title	Assignment I
Purpose of the assessment (with ULO Mapping)	Assess student's ability to develop algorithmic solutions to programming problems using Python language.
Weight	15 % of the total assessments
Total Marks	15
Word limit	N/A
Due Date	Week 8 (14, May 2019 11:55pm)
Submission Guidelines	• There are five questions in this assignment which require you to write and submit five Python scripts. Please save each script in two different formats: .py and .txt prior to submission. For example, for the first question you need to submit triangle.py and triangle.txt. • Code must be appropriately commented. Make sure to add comments at each segment of your code to explain what it does. • 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, 14 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.

Assignment I Specifications

Purpose:

This assignment evaluates your understanding of basic programming principles using Python language. In particular, it assesses your ability to develop algorithms to solve simple problems, successfully develop and run python programs, and your ability to write meaningful comments when necessary.

Marking criteria

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

1. Write a program **triangle.py** to determine whether a triangle is a 'right triangle'. A right triangle is a triangle where the square of one side equals the sum of the squares of the other two sides. Your program should prompt the user to enter the three sides of a triangle – one at a time – and then print a statement indicating whether or not the triangle is right. (3 marks)

2. Bit shifting is a process where the bits of a given string are moved to the left or to the right. For instance, the bits in the string **1110** becomes **1011** when shifted two places to the left. Notice that the leftmost two bits are wrapped around to the right side of the string in this example. Write a program **shift.py** which take a bit string as an input. The program prompts the user to enter a string of binary bits. The program allows the user to determine the direction of shifting by entering L for left shift or R for right shift. Once the bits string is entered and the shift direction is specified, the program shifts the bits two places either to the left or to the right according to the user choice. Finally, the program prints the resulting string. (3 marks)

3. The mathematical value of $\boldsymbol{\pi}$ can be approximated using the following equation:

   $\pi/4 = 1 – 1/3 + 1/5 – 1/7 + . . .$

   The higher number of iterations, the more accurate the approximation is. For instance, and iteration of two ($\pi/4 = 1 – 1/3$) yields a value 2.66 of for π, while an iteration of 100 yield a value of 3.13. Write a program **pie.py** that prompts the user to enter the number of iterations in this method of approximation and prints out the resulting value of $\boldsymbol{\pi}$**.** (3 marks)

4. Company **A** keeps a record of its employee payments in a text file. Each line in the file is formatted as follows:

   <first name>   <hourly rate>   <hours worked>

   Write a program **wage.py** that prompts the user to enter a filename as an input and prints a report of the payments made to employees. For each employee, the total payment is calculated by multiplying their hourly payment rate by the number of hours they worked. The report must be formatted in a tabular format with the above header. (3 marks)

5. Design a game where the player rolls a couple of dices. The rules of this game are as follows: if the total dots of the two dices add up to seven, the player wins four dollars. For any other value, the player loses one dollar. The casino claims that there are many ways for the players to wine. For example, if the dots of the two dices are (1, 6), (2, 5) … (5,2), (6, 1). In reality, there is a very little chance for anyone to win this game. Your task is to write a program **dice.py** which demonstrates that playing the game is pointless. Your program should prompt the user to enter the amount of money they wish to invest in this game, and play the game until this amount is depleted (becomes zero). At this point, the program prints a message that shows the number of rolls it took to break the player and the maximum amount of money the player won throughout the lifecycle of the game. (hint: you may need to import the **random** module and **randint** function to solve this problem). (3 marks)