**Parables and Pythons – Programming Contest**

The following are a list of programming challenges for the Parables and Pythons class held on 28 November 2018.

**Rules for the programming contest:**

* You can attempt as many challenges as you like.
* Each challenge has a maximum point total – those with higher values are more complicated.
* You can earn partial credit for each challenge if it is partially correct, but there must be no syntax errors.
* You may not use the Internet to search for solutions.
* You may ask for explanation of technical details, but you may not ask any Python questions.
* The student with the highest point total wins.
* Name your solution to match the number of the challenge. For example, challenge\_01.py or solution\_03.py.
* You must email your solutions to me by the end of class (drcharlesbell@gmail.com).
* Winner will be notified by email.

**Easy Challenges (1 Point)**

1. Write a Python program that takes a sequence of numbers and determines if all the numbers are different from each other.
2. Write a Python program to create all possible strings by using 'a', 'e', 'i', 'o', 'u'. Use the characters exactly once.
3. Write a Python program to remove and print every third number from a list of numbers until the list becomes empty.
4. Write a Python program to find the number of divisors of a given integer is even or odd.
5. Write a Python script to concatenate following dictionaries to create a new one.
6. We have a function that controls two monkeys, a and b, and the parameters a\_smile and b\_smile indicate if each is smiling. We are in trouble if they are both smiling or if neither of them is smiling. Return True if we are in trouble. Solve the challenge by completing this code:

def monkey\_trouble(a\_smile, b\_smile):

# Complete this code

print("Are we in trouble if:")

print("Both A and B are smiling?", monkey\_trouble(True, True))

print("Neither A nor B are smiling?", monkey\_trouble(False, False))

print("Only A is smiling?", monkey\_trouble(True, False))

print("Only B is smiling?", monkey\_trouble(False, True))

**Moderate Challenges (5 Points)**

1. Write a Python program to print a long text, convert the string to a list and print all the words and their frequencies.
2. Write a Python program to check the sum of three elements (each from an array) from three arrays is equal to a target value. Print all those three-element combinations.

Sample data:

/\*

X = [10, 20, 20, 20]

Y = [10, 20, 30, 40]

Z = [10, 30, 40, 20]

target = 70

\*/

1. Write a Python program to get the third side of right angled triangle from two given sides.
2. Write a Python script to sort (ascending and descending) a dictionary by value.
3. Write a Python program to print all unique values in a dictionary. Sample Data : [{"V":"S001"}, {"V": "S002"}, {"VI": "S001"}, {"VI": "S005"}, {"VII":"S005"}, {"V":"S009"},{"VIII":"S007"}] Expected Output : Unique Values: {'S005', 'S002', 'S007', 'S001', 'S009'}
4. Make a two-player Rock-Paper-Scissors game. (Hint: Ask for player plays (using input), compare them, print out a message of congratulations to the winner, and ask if the players want to start a new game). Use a function named find\_winner() that accepts two parameters; the answer from player 1 and the answer from player 2.

Remember the rules:

Rock beats scissors

Scissors beats paper

Paper beats rock

**Difficult Challenges (20 points)**

1. Write a Python program to display some information about the OS where the script is running. Sort the variables in alphabetical order. Hint: import platform.
2. Write a Python program to add two positive integers without using the '+' operator. Note: Use bit wise operations to add two numbers.
3. Write a Python program to get all strobogrammatic numbers that are of length n. A strobogrammatic number is a number whose numeral is rotationally symmetric, so that it appears the same when rotated 180 degrees. In other words, the numeral looks the same right-side up and upside down (e.g., 69, 96, 1001). For example, Given n = 2, return ["11", "69", "88", "96"]. Given n = 3, return ['818', '111', '916', '619', '808', '101', '906', '609', '888', '181', '986', '689']
4. Write a Python program to create a dictionary from a string. Note: Track the count of the letters from the string. Sample string: 'w3resource'. Expected output: {'3': 1, 's': 1, 'r': 2, 'u': 1, 'w': 1, 'c': 1, 'e': 2, 'o': 1}
5. Write a Python program which accepts the radius of a circle from the user and compute the area. Sample Input: r = 1.1. Sample Output: Area = 3.8013271108436504
6. Write a Python program to calculate number of days between two dates. Sample dates : (2014, 7, 2), (2014, 7, 11). Expected output: 9 days
7. Write a program that asks the user how many Fibonacci numbers to generate and then generates them. Take this opportunity to think about how you can use functions. Make sure to ask the user to enter the number of numbers in the sequence to generate.(Hint: The Fibonacci sequence is a sequence of numbers where the next number in the sequence is the sum of the previous two numbers in the sequence. The sequence looks like this: 1, 1, 2, 3, 5, 8, 13, …)

**Insanely Difficult Challenges (100 points)**

1. Write a Python program that plays checkers. Input should be in the form of grid coordinates for starting position (to select the checker) and destination. The program must validate both the piece and the move. Standard checkers rules apply.