Year 11 to 12 Computer Science

Transition Work 2018

- In this document are 7 programming challenges
- Implement the solutions as separate files in Python
- Bring your solutions in electronic format (on a memory stick or online storage) at the beginning of Year 12

Must: Complete any 5 of the given problems

Should: Complete all the given problems

Could: Complete an online tutorial though codecademy to learn the basics of another language of your choice.

Problem 1: Football Pools

Introduction

The pools is a game based on predicting the outcome of football matches.
The bet is placed on a list of 15 matches, normally 10 of Premier League and 5 of Championship. The possible values for each of the matches are 1 (meaning that the home team wins), X (meaning tie) or 2 (meaning that the visiting team wins).
You have been requested to develop a system to evaluate the number of hits of every bet.

Input

The input will be two strings. The first string contains the 15 results of the matches while the second string represents a single bet that must be evaluated.

```
X111XXX21X21222
11XXX2XXX221111
```

Output

The program must output the number of hits for current bet.

```
5
```

Problem 2: The Arrival

Introduction

The YETI (Yet searching for Extra Terrestrial Intelligence) is receiving a signal from a remote solar system.
The message is a long sequence containing characters and numbers and as you may imagine, it seems impossible to decipher...
But you have been asked to isolate the signal by splitting it into two sequences -- one with characters and another with numbers – in order to continue the investigation. Let's do it!

Input

The input will be a string containing characters and numbers with no spaces.

```
0HELLO1WORLD2!3WE4CAME5FROM6A7GALAXY8FAR9FAR0AWAY
```

Output

The program must output two strings. The first one contains all the characters except numbers and the second contains only numbers.

```
HELLOWORLD!WECAMEFROMAGALAXYFARFARAWAY
01234567890
```

Problem 3: Bike Race

Introduction

Next month, there will be a bike race in Byker and the race organisers have asked you to create an application to calculate the final classification. It will be a timed race, the competitors will start at different times, and the important thing is the time they take to complete the circuit.

To know how long a competitor takes, there will be two sensors that will save both the timestamp the cyclist starts and the time he crosses the finish line. This data will be sent to your application. The timestamps are epoch time, meaning that each timestamp will save the seconds elapsed since 1970.

Input

The input will be a sequence of all the competitors names with their initial and their final timestamp value, ending with a #.

```
Bernard 1483519913 1483523201
Joel 1483520440 1483523498
Lucas 1483521040 1483524254
#
```

Output

The output should be a list of all the competitors, each of them with their final result expressed in seconds, and sorted by first to last position.

```
Joel 3058
Lucas 3214
Bernard 3288
```

Problem 4: The Apple Fall

Introduction

The Interplanetary Space Organization is studying the gravity in different planets with the most advanced orbital telescope by checking how long it takes for an apple to fall from the top of the apple tree to the ground on each planet.
The data collected by the scientists for each planet consist of the height of the tree and the time it took the apple to land on the ground.
Your mission is to find out which one of the planets in this research has the strongest gravity.

$$h = \frac{1}{2} g t^2$$

Input format:

[Number of planets in this research (N)]
[Height of the tree in meters] [Time to fall in seconds] <-- One line for each planet numbered 1, 2, 3,...,N

Output format:

Planet number [Number of planet] has the most gravity.

Input

2
3.25 2
20 2.474

Output

Planet number 2 has the most gravity.

Problem 5: Magic Squares

Introduction

A magic square, is a n x n square grid filled with distinct positive integers in the range 1, 2, …, n^2 such that each cell contains a different integer and the sum of the integers in each row, column and main diagonal is equal. This sum is called the magic constant of the magic square. Even though magic square do not have a known application, and they belong to the recreational mathematics space, they have a long history, dating back to at least 650 BC in China. Many times, this squares have acquired magical or mythical significance, and have appeared as symbols in works of art. In Europe, one of the most famous magic squares is the Albert Dürer order-4 magic square, immortalized in his 1514 engraving Melencolia I:

16	3	2	13
5	10	11	8
9	6	7	12
4	15	14	1

Write a program to check if a given square is magic or not.

Input

A series of rows with the corresponding columns of the magic square. As the order of the square might be variable the finish of the input will be marked with the character '#'.

Example 1

```
8 3 4
1 5 9
6 7 2
#
```

Example 2
8 13 2 11
1 12 7 14
15 6 16 4
10 3 9 5
#

Output

A string reporting whether the input is or is not a magic square.

Example 1
```
This is a magic square
```

Example 2
```
This is not a magic square
```

Problem 6: Roman Calculator

Introduction

Maximilian lives in the Roman Empire and he has a grocery.
The business goes very well but he has a little problem: all the prices are in roman numerals and making sums with them is a painfully task.
Thank to Mercury, patron god of commerce, you are his friend, a well-known programmer in the empire.
Help to Maximilian making a roman calculator for attend customers faster.

Roman Numerals	
I	1
V	5
X	10
L	50
C	100
D	500
M	1000

Input

The input will be a sequence of roman numerals ended with a +

```
I
II
IV
VI
+
```

Output

The output will be the sum of the given roman numerals, also in roman
The sum always will be less than 4000

```
XIII
```

Problem 7: Goldbach's conjecture

Introduction

In 1742 the Prussian mathematician Christian Goldbach wrote a letter to his friend Leonhard Euler where he stated that:
"Any even number greater than 2 can be written as the sum of two prime numbers"
This statement is known as Goldbach's conjecture. Since then, mathematicians have been able to find such a pair of prime numbers for any even number greater than 2 that they've considered. It has been checked using computers for even numbers up to 4 × 1018. Up to date nobody has been able to prove that this result holds for all even numbers. So, the conjecture remains unproven despite considerable effort.

Input

The input will be an even number greater than 4 and lesser than 5000. For example:

```
128
```

Output

The output of the program is a list of pair of prime numbers that sum up the number provided in the input.
The list is ordered in increasing order considering the value of the first prime addend.
This is the output for the previous example:

```
19 + 109
31 + 97
61 + 67
```