Introduction to Python Weeks 1&2

Exercises:

- a) Write a program that prints Hello World! onto the screen
- b) Write a program that asks the user for their name and then greets them
- c) Write a program that asks the user for the radius of a circle and then prints the area of the circle on the screen

```
Hint1: pi = 3.14
```

Hint2: the variable stored from input() is a string, convert
to an integer with int()

- d) Write a program that asks the user for a number n and prints the sum of the numbers 1 to n
- e) Write a program that prints a multiplication table for numbers up to 12.
- f) Write a program that asks for a number and then prints the square root of that number, if it is a square number.
- g) Write a program that asks the user to input a year and tells us if it is a leap year
 - I. A year may be a leap year if it is evenly divisible by 4.
 - II. Years that are divisible by 100 cannot be leap years unless they are also divisible by 400.
- h) Write a guessing game where the user has to guess a secret number. After every guess the program tells the user whether their number was too large or too small. At the end the number of tries needed should be printed. It counts only as one try if they input the same number multiple times consecutively.
- i) Write a program that concatenates two lists. E.g., [a,b,c], [1,2,3]→ [a,b,c,1,2,3]
- j) Write a function that combines two lists by alternatingly taking elements, e.g. [a,b,c], [1,2,3] → [a,1,b,2,c,3].

- k) Write a program that creates a list of random length (up to 100 elements) of random integers between -100 and 100.
- 1) Using your solution to k), write a program that:
 - I. Prints out the sum of all elements of the list
 - II. Prints out the largest element of the list
 - III. Prints out the product of every third element of the list
- m) Create two random lists, a and b, both 20 elements long. Create a new list, whose i^{th} element is a 1 if the i^{th} element of b exists anywhere in a, and is 0 otherwise.
- n) Write a program asks the user for some text, and then reverses that text.

Challenges

o) The four adjacent digits in the 1000-digit number below that have the greatest product are $9 \times 9 \times 8 \times 9 = 5832$.

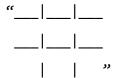
 $73167176531330624919225119674426574742355349194934969835203127745063262395\\ 78318016984801869478851843858615607891129494954595017379583319528532088055\\ 11125406987471585238630507156932909632952274430435576689664895044524452316\\ 17318564030987111217223831136222989342338030813533627661428280644448664523\\ 87493035890729629049156044077239071381051585930796086670172427121883998797\\ 90879227492190169972088809377665727333001053367881220235421809751254540594\\ 75224352584907711670556013604839586446706324415722155397536978179778461740\\ 64955149290862569321978468622482839722413756570560574902614079729686524145\\ 35100474821663704844031998900088952434506585412275886668811642717147992444\\ 29282308634656748139191231628245861786645835912456652947654568284891288314\\ 26076900422421902267105562632111110937054421750694165896040807198403850962\\ 45544436298123098787992724428490918884580156166097919133875499200524063689\\ 91256071760605886116467109405077541002256983155200055935729725716362695618\\ 82670428252483600823257530420752963450$

Find the thirteen adjacent digits in the 1000-digit number that have the greatest product. What is the value of this product?

p) Write a program that asks the user for some text and then asks for a number. The text should then be put into a secret code, by shifting the letters either to the left or right by the number given e.g. if asked for 3, the letter a->d, b->e, z->c etc. Give a friend a secret message and challenge them to decode it. Try using the rfind() function, to return the position of a letter in a string.

Hint:

- q) Create a program that that plays a game of noughts and crosses with the user. Try breaking the problem down into the following steps:
 - I. Print out the noughts and crosses board:



- II. Ask the user where they would like to play (row and column), and print the board with an X at that location
- III. Create some way of storing the played locations, so that the same space can't be played on twice.
- IV. Have the computer place an O at some point (this can be random, or try implement a strategy)
- V. Repeat the steps above until the game is over, you will need to implement some method of checking whether the user or computer have won.