# IC152: Assignment 4
# Loops Continued, Lists, and Problems involving Lists

In this assignment, continue with writing loops, use lists, list methods, and solve some problems involving lists. You have to write one python file for each question. So, in total there will be 4 python files to submit for this assignment.

If you are solving this assignment in the A11's PC Lab: Keep Fn + F9 pressed during the start of your machine (do not repeatedly press, keep it continuously pressed), and then select the second option with "ubuntu". Please check if you are able to log in to moodle, or else change the machine.

Only one member needs to submit the solutions with a pdf on learnings of all group members. This is your last practice assignment. From the next lab you will be evaluated based on viva.

**Problem 1:** Loops Continued.
Write a common python file "q1Loopscntd.py" and prompt the user with "q1 part a input (integer): ", "q1 part b input (list): " and so on for different parts

a) Define a function that takes an integer n as input and returns a list of all prime numbers less than or equal to n. Implement using loops and lists for storage.
   Example: For n = 10, the output should be [2, 3, 5, 7].

b) Define a function to count the number of occurrences of each element in any custom list. For the sake of the question, assume the elements to be numbers. The output should be a nested list, where every element is a list of the format [number,occurrence]. If the list is empty, return -1.
For eg: [1,2,3,1,1,2,1,1] : ans = [[1,5],[2,2],[3,1]]

c) Write a program that finds two distinct elements in a list that sum up to a given target number. If such a pair exists, return the pair as a tuple; otherwise, return -1.
Example: For the input list [2, 4, 3, 5, 7] and target = 9, the output should be (4, 5).

**Algorithms for Problem 1:**

a) **Task:** Define a function that takes an integer n as input and returns a list of all prime numbers less than or equal to n. Implement using loops and lists for storage.
Example: For n = 10, the output should be [2, 3, 5, 7].

**Constraint:** all variables are integers, and a list of integers is needed

**Input:** a number n

**Output:** list of all prime numbers less than or equal to n as return of function.

**Algorithm:**

Take n as input from the user
Pass it to a function with following statements:-
Initialise a variable i with 1 value
Create an empty list, say l.
Repeat n times the following statements:
Check all factors of i starting from 1 which are <=i
If i has exactly two distinct factors, then increment i to the list l.
Increment i by one

**Problem 2:** List Operations.
Write a common python file “q2ListOperations.py” and prompt the user with “q2 part a input (list): ”, “q2 part b input (integer): ” and so on for different parts.

a) Write a program that sorts the input list according to their distances from the mean of the list. For example, [5,10,20,25,40] will become [20,25,10,5,40]. If two numbers are at the same distance, the smaller one must be placed first.

b) Write a function that rotates a list to the left by k positions, where k is provided by the user.Example: For input list [1, 2, 3, 4, 5] and k = 2, the output should be [3, 4, 5, 1, 2].

c) Write a function that takes a list of numbers and returns a new list where each element is the cumulative product of the elements up to that point in the original list. Example: For the input [1, 2, 3, 4], the output should be [1, 2, 6, 24].

d) Write a program that finds all occurrences of a given element k in a list and returns a list of all the indices where k appears. If k

does not exist in the list, return an empty list. Example: For the input [10, 20, 10, 30, 10] and k = 10, the output should be [0, 2, 4].

## Algorithms for Problem 2:

a) **Task:** Write a program that sorts the input list according to their distances from the mean of the list. For example, [5,10,20,25,40] will become [20,25,10,5,40]. If two numbers are at the same distance, the smaller one must be placed first.

**Constraint:** A finite length of list.

**Input:** A list

**Output:** A list sorted in the order of distance from x to mean, if this distance is same for two numbers, the lower number value will be placed first

**Algorithm:**
Define a list and take the input for it.
If the list is empty, return it immediately.
Define a function and pass given list to it with the following statements:
Calculate the mean of the list
Create an empty dictionary to store distance and corresponding values
For each element over the list
Calculate the absolute distance from the mean
If distance is not already a key in dictionary
Create a new entry in the dictionary with key distance and value as a list containing element
Else
Append element to the list of values for corresponding distance in the dictionary
Create an empty list sorted_list
For each key(distance) in the dictionary
Sort the list of values for that distance in ascending order
Append each value in the sorted_list list
Print the sorted_list list

a) **Task:** Write a function that rotates a list to the left by k positions, where k is provided by the user.Example: For input list [1, 2, 3, 4, 5] and k = 2, the output should be [3, 4, 5, 1, 2].

**Constraint:** A finite length of list and a positive integer k.

**Input:** A list, and a number k.

**Output:** A rotated list to the left by k positions.

**Algorithm:**
Define a list and take the input for it.
Take ‘k’ as input from the user for shifting the list to the left by k.
If the list is empty, return the empty list immediately.
Define a function & pass given list & variable k to it with the following statements:-

Check if k is larger than the length of list. For handling this use k mod (length of list), it wraps around correctly.

Perform rotation, slice the list into two parts-

The part from index k to the end of the list.

The part from the start of the list to index k.

Concatenate above two parts to form the left rotated list and return.
Print the o/p rotated list.

c) **Task:** Define a function that takes a list of numbers and returns a new list where each element is the cumulative product of the elements up to that point in the original list.

Example: For input [1, 2, 3, 4], the output should be [1, 2, 6, 24].

**Constraint:** A finite list of integers inputted and outputted as

demanded.

**Input:** A finite integer and a finite list of integers of inputted size.

**Output:** A list of the cumulative product of the elements up to that point in the original list.

**Algorithm:**

Take n as input from the user for the size of the list.

Define a list, take the input of each integer and store it in the same list.

Define a function and pass the list, m to a function with following statements:-

Initialise a empty list, l to store the cumulative product upto the current element.

Repeat for size of the list m for the following statements:

Store the first element of list m to list l.

For other elements, multiply the preceding element of l and current element of m to store the result in the current place of the list m.

Increment to the next element of l.

D. **Task:** Write a program that finds all occurrences of a given element k in a list and returns a list of all the indices where k appears. If k does not exist in the list, return an empty list.

**Example:** For the input list [10, 20, 10, 30, 10] and k = 10, the output should be [0, 2, 4].

**Constraint:** The list can have duplicate elements.

If k is not found in the list, the output should be an empty list.

**Input:** A list of integers.

An integer k to search for in the list.

**Output:** A list of all indices where k appears in the input list.

If k does not appear, return an empty list.

**Algorithm:**

Take a list of integers and the integer k to search for as inputs from the user.

Define a function that takes the list and k as input.

Initialize an empty list to store the indices where k appears.

Loop through the list. For each element, check if it is equal to k.

If the element is equal to k, append the current index to the list of indices.

Return the list of indices. If no occurrences of k are found, return the empty list.

**Problem 3:** Polynomial functions using lists.
Write a common python file “q3Polynomial.py” and prompt the user with “q3 part a input (list): ”, “q3 part b input (integer): ” and so on for different parts.

a. Take the coefficients of the polynomial function from the user as a list. Now define a function to return the coefficient of the given degree provided by the user = n;
   Example: input: [2, 5, 0, 1] , n = 2 # represents $2x^3 + 5x^2 + 0x + 1$
   Output: 5
b. Given the polynomial function: $f(x) = 4x^3 - 6x^2 + 0x - 1$
   Write a program that takes the input of x from the user and returns the value of the polynomial for the given value of x.

c. Take input of two polynomial functions from the user in the form of lists as above. Now Write a program to evaluate the sum of the two polynomial functions and print the final polynomial.
Example input: f1(x) = $2x^3 + 3x^2 - 10$;
f2(x) = $4x^5 + 1x^3 + 2x + 1$
Output: "f(x) = $4x^5 + 3x^3 + 3x^2 + 2x - 9$"

**Algorithms for Problem 3:**

a. **Task:** We are given a list of coefficients of a polynomial. We want to find the coefficient of degree n where n is provided by the user.

**Constraints:** All input must be integer i.e. every element of the list and value of n must be an integer. Also n must be positive.

**Input:** A list containing coefficients of a polynomial, and a number n.

**Output:** The value of the coefficient of the degree n in the polynomial.

**Algorithm:**

Take a list containing integers as an input from the user.
Also take a integer as an input from the
User.
If n>=len(L): Return 0
Else: Return the value L[-(n+1)]

b. **Task:** Given the polynomial function: f(x) = $4x^3 - 6x^2 + 0x - 1$Write a program that takes the input of x from the user and returns the value of the polynomial for the given value of x.

**Constraints:** All input either an integer or a floating-point number.

**Input:** A number x

**Output:** The value of the polynomial for the given x

**Algorithm:**

Take x as input from the user
Pass it to a function with following statements:-
Initialise a variable to store the value of the polynomial.
Calculate the value of the polynomial using the formula
f(x)= 4x^3 - 6x^2 + 0x - 1
Use the following operations:
Calculate 4x^3 store result in a variable
Calculate 6x^2 store result in another variable
Calculate 0x(which always gives zero)
Subtract the second result from the first, then subtract 1 from the result.
return the Calculated result after all the operation

c. **Task:** We need to write a program to evaluate the sum of the two polynomial functions and print the final polynomial.

**Constraints:** Both the lists should contain numeric values.

**Input:** 2 lists that contain the coefficients of 2 polynomials respectively.

**Output:** A polynomial formed by the summation of input polynomials.

**Algorithm:**

**(assuming 1st index to be 1 (not 0))**

1. Take 2 lists containing numeric data as inputs from the user (let's say list 1 and list 2).
2. Now reverse both the list, say new lists are l1 and l2.
3. Check the length of both the lists (say n1,n2) and run a for loop of range equal to the value of length of the smaller list (1 to say n1).
4. Make a new list (say l) in which the value at i th index will be the sum of the value of l1 at ith index and value of l2 at ith index.
5. Now for (n2-n1) append the elements of l2 from (n1+1) th index to (n2-n1) th index.
6. Reverse the final list (l).
7. Print the output in the required form by running a loop on the list in reverse order and adding x,x^2,x^3..... Strings in concatenation with the coefficient values.

**Problem 4:** 1D and 2D Lists

a. Write a user defined function in python that takes a list with name "numList" as input argument and finds the maximum of three consecutive numbers, stores them in a list and prints it. For e.g numList = [1,34,3,2,5]

[ 1, 34, 3, 2, 5 ], the max of first 3 elements is 34
[1, 34, 3, 2, 5], the max of these 3 elements is 34
[1, 34, 3, 2, 5], the max of last 3 elements is 5

**Output**: The function should return the list [34, 34, 5]. You can print this after the function call.

**IMPORTANT**: You will first need to convert string (e.g. "[1,34,3,2,5]" to a list, use the inbuilt eval() function for the same).

b. Write a program to take the input of a 2D matrix from the user. Start with taking input for a number of rows and a number of columns in the matrix. Then input the matrix row by row.
Print this matrix in a matrix format.

c. Create a 2D list using numList as num2DList, with values from numList and dimensions nxm (rows=n, columns=m), e.g n = 3, m = 3 and numList = [1,4,2,9,11,10,19,5,12], the num2DList will look like
[[1, 4, 2], [9, 11, 10], [19, 5, 12]], now use the function defined above in problem 5 part a to determine the maximum of three consecutive numbers for each row, store them in a 2D list and return it.
For e.g num2Dlist = [[1, 4, 2], [9, 11, 10], [19, 5, 12]] so the output should be [[4], [11], [19]]. In case of m<3 return an empty list.

**Algorithms for Problem 4:**

**a)**

**b) Task:** Inputting data of n x m numbers in the form of a 2d matrix and printing the matrix in matrix (tabular) form

**Constraint:** number of rows and columns should be natural numbers

**Input:** n rows containing m elements each

**Output:** The matrix in matrix form

**Algorithm:**

1. Input the number of rows and column as n and m
2. Create an empty 2d list l
3. Repeat the following n times:
    Create another empty list 'temp'
    Repeat the following m times:
        Input an integer
        Append the integer to temp
    Append temp to l

Now, your data is stored as a list of lists. If you directly print l, it will be printed as: [[2,3,4],[5,6,7],[8,9,0]]. But we want it as:

```
2 3 4
5 6 7
8 9 0
```

4. Repeat the following n times with iterator i(say):
    Repeat the following m times with iterator j(say):
        Print the jth element of the ith row and a space
    Print a blank space

**c)**

**Extra/Bonus question:** Write a program to find roots of a given polynomial in continuation to problem 3.

Create the folder having your python files, with name having your roll number followed by “_assignment4” (don’t use inverted commas in folder name), compress the folder with .zip extension and submit it on moodle.

Make sure that you delete all your files from the lab PC/Laptop, and shut it down before you leave.