ICT1054

Data Structures and Algorithms

Icon

Description automatically generated

University of Malta

B.Sc. IT (Hons.) Computing and Business

Luigi Naudi

0249603(L)

Contents:

Documentation:

Plagiarism Declaration………………………………………...…………………………………………3

Statement of Completion……………………………………………………………………………...…4

Question1......................................................................................................................................5

Question 2.....................................................................................................................................6

Question 3.....................................................................................................................................7

Question 4.....................................................................................................................................8

Question 5...................................................................................................................................10

Question 6...................................................................................................................................12

Question 7...................................................................................................................................13

Question 8...................................................................................................................................15

Question 9...................................................................................................................................16

Question 10.................................................................................................................................17

Question 11.................................................................................................................................18

Question 12.................................................................................................................................19

References..................................................................................................................................21

Source Code Listing:

Question1 and 2..........................................................................................................................22

Question 3...................................................................................................................................24

Question 4...................................................................................................................................25

Question 5...................................................................................................................................26

Question 6...................................................................................................................................27

Question 7...................................................................................................................................28

Question 8...................................................................................................................................31

Question 9...................................................................................................................................31

Question 10.................................................................................................................................32

Question 11.................................................................................................................................32

Question 12.................................................................................................................................33

Insert signed pladgiarism form here before uploading

Statement of Completion:



Question 1 - Attempted and Works Well

Question 2 – Attempted and Works Well

Question 3 – Attempted and Works Well

Question 4 – Attempted and Works Well

Question 5 – Completed but has some bugs:

* Crashes when coming across invalid char or a non RPN expression

Question 6 – Attempted and Works Well

Question 7 – Attempted and Works Well

* Ran into difficulty displaying BST

Question 8 – Attempted and Works Well

Question 9 – Attempted and Works Well

Question 10 – Attempted and Works Well

Question 11 – Attempted and Works Well

Question 12 – Attempted and Works Well



**Question 1**

References: [1] [2]

A picture containing calendar

Description automatically generatedThe program creates two arrays of random sizes 256 and 300 containing randomly generated numbers between 0 and 1024.Array A is sorted using Shell sort, whilst array B is sorted using Quick sort. The output for Array A shows the unsorted array, and the sorted array after Shell sort has been implemented. The output for array B also shows the unsorted array, and the sorted array after Quick sort has been implemented.

**Question 2**

Since Question 2 is a continuation of Question 1, it was made sure that it works well before moving on. An empty array C was created as well as two pointers, one for array A and one for array B. The pointers are used in the algorithm to decide which element from both arrays is smallest. The smallest element is added to array C whilst the pointer of the smaller element is incremented, and compared to the other pointer once again, and so on. The pointers allow for the merged array C to be sorted from the beginning of the populating process. Looking at the previous results of array A and B it is evident that the merging was is successful and sorted as expected.

A picture containing text

Description automatically generated

**Question 3**

There are no extreme points in a sorted array since element (x) will always be larger than element (x-1) and less than element (x+1), thus, to test this question I had to use both an unsorted array and a sorted array. It was chosen not to randomly generate the numbers of the array since the likelihood of it being sorted is way less than that of it being sorted.

Text

Description automatically generatedWhen using a sorted array, the output shows that it has been recognised that there are no extreme points in the array:

On the other hand, when using an unsorted array, the output indicated clearly that there are extreme points in the array and can identify them clearly.

Text

Description automatically generated

This indicates that the algorithm can cater for both types of arrays and has an expected and therefore correct output.

**Question 4**

For this Question an array of 30 random numbers is generated. Then, the product of each possible pair without repetition is calculated and placed into another array in the format [element A, element B, their product]. Two variables store and comp hold the products of two elements in the array and are compared. If they are equal, they are pushed to another array.

The problem was duplicate elements in the pairs array, which lead to duplicate outputs in the end. To solve this, the array is pushed into a hash set (then converted back into an array) which would filter and remove duplicate elements from the array.

The output size differs from size due to the array being randomly generated. All scenarios were tested such as a no two-pair array, and an array with many duplicates:

No two-pair array output:

Text

Description automatically generated

Duplicate filled array:

Graphical user interface, text

Description automatically generated

Random Generated Array:

Text

Description automatically generated

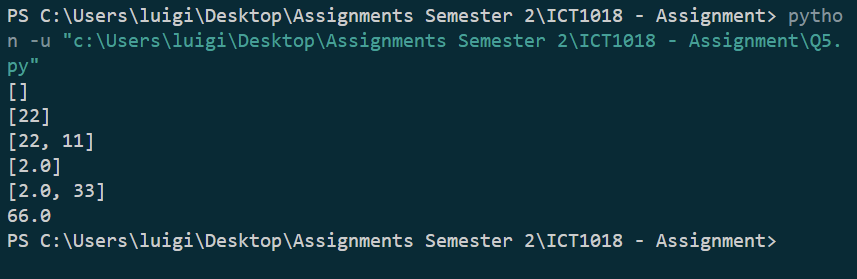
Apart from these, tests were made with smaller two-pair arrays to check if the correctness of output.

Text

Description automatically generated

**Question 5**

This Question was tested with different RPN’s of varying sizes. Result was as expected:

Diagram, schematic

Description automatically generatedA picture containing text, wall

Description automatically generatedText, email

Description automatically generated

It was also tested using an incorrect input ‘b’ which leads the program to crash since it is not programmed to handle anything other than integers and simple arithmetic operators:

Text

Description automatically generated

Diagram, schematic

Description automatically generatedFinally, it was tested with a previous mathematic sum used, but in PN format instead:

Text

Description automatically generated

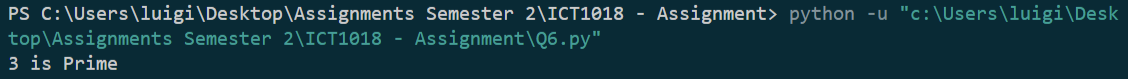
As expected, the program crashed here too. This is because the calculator is only programmed to handle RPN expressions

**Question 6**

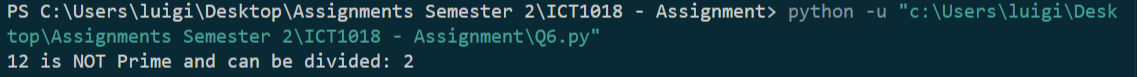
References: [3]

The first Question was tested with a prime number (3) and a non-prime number (12). After verifying that it works, A random number was used to test and clarify that the algorithm produces the expected output.

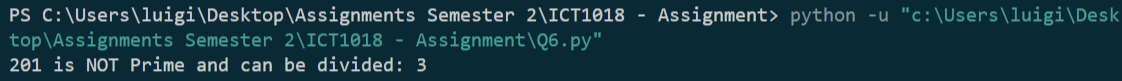
Prime Number Test:



Non-Prime Number Test:



Random Number Test:



Graphical user interface, text

Description automatically generatedFor the Second Question testing, inputs from limits between 10 – 60 were randomly picked. Naturally if the input 60 were to produce the correct output, all limits under that number should produce the expected result up to that limit also. Nonetheless all the limits were tested for certainty and the outputs were as expected.

**Question 7**

References: [4]

For this question the requirements were clearly listed to plan. Displaying the BST visually was a problem I ran into, thus testing if a proper tree could be built was difficult.

Text

Description automatically generated with medium confidenceInitially it was checked to see if the algorithm could properly handle a root node and a left and right node for it:

After this, an algorithm to visually represent the BST was created. It had some flaws at first due to recursion and the .format method disagreeing with each other and producing an expected but unwanted output.

With three simple input it works well:

Graphical user interface, text

Description automatically generated

As inputs grew, the algorithm showed its flaws and was not displaying the BST the way it was intended:

A computer screen capture

Description automatically generated with medium confidence

The problem was that the .format spacing was not being changed after each method call, thus the numbers after the second level of the BST being printed out underneath each other. Apart from that, as seen above, 101 is placed under the left side of the subtree, this is because Node 101 is still 150’s leftNode, but the display algorithm has no way of realizing that to place it under 150.

Although the above can easily be implemented, still, the program would have no way to indent for each level and cater for each subtree. The algorithm was scrapped but included in the code as a comment.

The code from line 48-84 was inspired by [4] and works well. It was tested using a range of outputs and produced results as expected:

Text

Description automatically generated

Text

Description automatically generated

**Question 8**

References: [5]

This Question’s accuracy is dependent on the number of iterations the algorithm goes through. To test, different numbers were inputted to calculate an approximation of their square root.

Text

Description automatically generated

The result was compared to an online calculator to check the accuracy and correctness of the output. The results were as expected.

For this question it was particularly interesting testing different iterations and noticing the change in accuracy of calculations:

Text

Description automatically generatedText

Description automatically generatedText

Description automatically generated

**Question 9**

For this question a similar approach to Q4 was taken using sets to locate repeated elements. It was tested using two different arrays, one with many repeated elements and another with no repeated elements.

Testing using an array with repeated elements. Expected output was:

[1, 2, 3, 6, 7, 8, 9, 11, 13, 51]

Result was correct and as expected

Text

Description automatically generated

Testing using an array with no repeated elements. The expected output was correct:

Text

Description automatically generated

**Question 10**

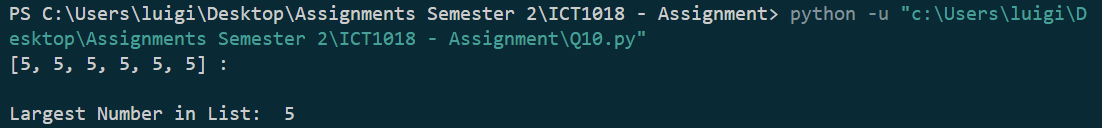
References: [6]

To test this a list of random numbers was used, with the largest number known, the expected output was 55, showing that the result below was correct:

Text

Description automatically generated

It was also tested using an array of the same number only, making the expected output to be 5, and as seen below, correct:



The last test checked if the program could handle negative numbers and 0, which as expected produced the correct result once again:

A screenshot of a computer

Description automatically generated with medium confidence

**Question 11**

References [7] [8], were used to obtain and implement the formulas for sin and cos.

This Question was tested using different values for n (terms of the expansion) as well as different values of x for cos(x) and sin(x). The outputs were compared to an online calculator: <https://www.wolframalpha.com/calculators/series-calculator>

Testing using n = 20 and setting x to 1.5

Text

Description automatically generated

Graphical user interface, application

Description automatically generatedGraphical user interface, application, Teams

Description automatically generatedConfirming result is therefore, as expected:

Text

Description automatically generatedTesting using n = 25 and setting x to 1.4

Graphical user interface, application, table

Description automatically generatedGraphical user interface, application

Description automatically generatedOnce again result was as expected:

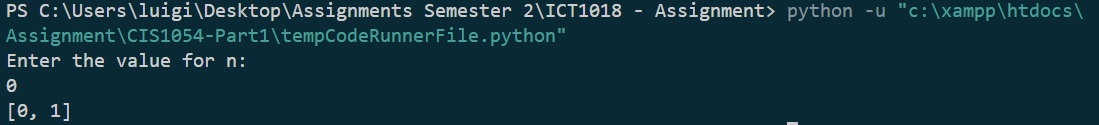
**Question 12**

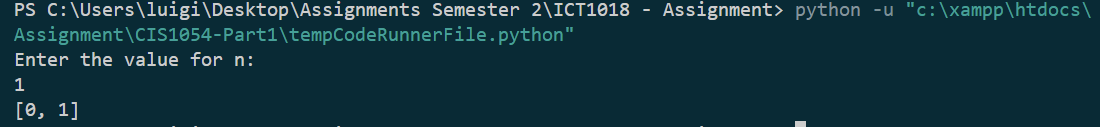
For this Question the code to generate the Fibonacci sequence was implemented. It was tested using a random number and the output was as expected.

Text

Description automatically generated

When input n was set to 0 or 1, the output was not as expected:





This was because the program had to way of detecting 0 and 1 and always printed the initial array [0,1]. The program was changed to cater for such inputs and output was as expected:

Graphical user interface, text

Description automatically generatedText

Description automatically generatedGraphical user interface, text

Description automatically generated

After the first part of the program was confirmed to function as expected the sum of each element of the Fibonacci sequence was implemented and tested using, once again, 0,1,2, and 12:

Text

Description automatically generated Text

Description automatically generated Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generatedIt was also tested with an invalid input such as a letter. Here the program crashes as it is not programmed to take anything other than an integer value

**References:**

[1]: “ShellSort.” *GeeksforGeeks*, 16 June 2014, www.geeksforgeeks.org/shellsort/

[2]: “Python Program for QuickSort.” *GeeksforGeeks*, 7 Jan. 2014,

www.geeksforgeeks.org/python-program-for-quicksort/

[3]: “Python Program for Sieve of Eratosthenes.” *GeeksforGeeks*, 27 July 2012, www.geeksforgeeks.org/python-program-for-sieve-of-eratosthenes/

[4]: “Print Binary Tree Level by Level in Python.” *Stack Overflow*, stackoverflow.com/questions/34012886/print-binary-tree-level-by-level-in-python.

[5]: “Find Root of a Number Using Newton’s Method.” *GeeksforGeeks*, 7 Feb. 2020, www.geeksforgeeks.org/find-root-of-a-number-using-newtons-method/#:~:text=Let%20N%20be%20any%20number. Accessed 1 May 2022.

[6]: “Recursion - Python: Recursive Function to Find the Largest Number in the List.” *Stack Overflow*, stackoverflow.com/questions/12711397/python-recursive-function-to-find-the-largest-number-in-the-list. Accessed 1 May 2022.

[7]: “Math - Series Expansion of Cos with Python.” *Stack Overflow*, stackoverflow.com/questions/42302023/series-expansion-of-cos-with-python. Accessed 1 May 2022.

[8]: “How to Calculate Taylor Series and Lewis Carrol Divisbilty Test in Python 3.5 without Using the Math Module.” *Stack Overflow*, stackoverflow.com/questions/38002712/how-to-calculate-taylor-series-and-lewis-carrol-divisbilty-test-in-python-3-5-wi. Accessed 1 May 2022.

**Source Code Listing**

Question 1 and 2:

import random as rd

from turtle import position

*#Two Arrays of Unequal size (256 and 300) containing randomly generated numbers between 0 and 1024*

A = [rd.randrange(0,1025)for i in range(256)]

B = [rd.randrange(0,1025)for i in range(300)]

C = []

*#Shell Sort Algorithm Method*

**def** shell\_Sort(array):

    length = len(array)

    gap = length//2

    while gap > 0:

        for i in range(gap, length):

            temp = array[i]

            j = i

            while j >= gap and array[j - gap] > temp:

                array[j] = array[j - gap]

                j -= gap

            array[j] = temp

        gap //= 2

*#Quick Sort Algorithm Method*

**def** split(array, low, high):

    idx\_small = (low-1)

    pivot = array[high]

    for i in range(low, high):

*# Current element = array[j]*

        if array[i] <= pivot:

*# Increment index of smaller element*

            idx\_small = idx\_small+1

            array[idx\_small], array[i] = array[i], array[idx\_small]

    array[idx\_small+1], array[high] = array[high], array[idx\_small+1]

    return (idx\_small+1)

**def** quick\_Sort(array, low, high):

    if len(array) == 1:

        return array

    if low < high:

*# Index for Split*

        spin = split(array, low, high)

*# Sorting elements before and after splitting them up*

        quick\_Sort(array, low, spin-1)

        quick\_Sort(array, spin+1, high)

*#Printing Array A before and After using Shell Sort*

print("\nArray A Before Shell Sort: \n " ,A)

shell\_Sort(A)

print("\nArray A After Shell Sort: \n " ,A)

*#Printing Array B before and After using Quick Sort*

print("\nArray B Before Quick Sort: \n " ,B)

h = len(B) - 1

quick\_Sort(B, 0 ,h)

print("\nArray B After Quick Sort: \n " ,B)

*#Method to merge arrays using pointers and loops*

**def** merge\_Arrays(A, B, C):

*#Setting the length of Array C*

    length = len(A) + len(B)

    pointer\_A = 0

    pointer\_B = 0

    d = 0

    while (length - 2 > d):

        d = pointer\_A + pointer\_B

        if (pointer\_A == len(A)-1):

            C.append(B[pointer\_B])

            pointer\_B += 1

        elif (pointer\_B == len(B)-1):

            C.append(A[pointer\_A])

            pointer\_A += 1

        elif (A[pointer\_A] <= B[pointer\_B]):

            C.append(A[pointer\_A])

            pointer\_A += 1

        else:

            C.append(B[pointer\_B])

            pointer\_B += 1

merge\_Arrays(A, B, C)

print("\nArray C: \n" ,C)

Question 3:

*#Extreme Points = Element is bigger/smaller than both previous and following elements*

A = [0, 5, 3, 6, 8, 7, 15, 9]

*#A = [2,5,7,9,23,53,67]*

*#Array to Store all Random Points*

extreme\_Points = []

size = len(A)

*#Printing the Array*

print("Array: \n" ,A)

for i in range (0,size):

    if (0 < i < size-1) and ((A[i-1] < A[i] > A[i+1]) or (A[i-1] > A[i] < A[i+1])):

        extreme\_Points.append(A[i])

if(extreme\_Points == []):

    print("No Extreme Elements are in the Array")

else:

    print("\nExtreme Elements:\n")

*#Printing the Extreme Elements in the Array*

    print(extreme\_Points)

Question 4:

import random as rd

*#Creating a Random Array of 30 Numbers*

Temp\_Set = set()

nums = [rd.randrange(1,124)for i in range(30)]

*#nums = [2,5]*

*#nums = [2,5,10,13,2,12,8,4,6]*

*#nums = [2,5,62,46,1,42,75,24,8,35,13,11,23,9,2,5,62,46,1,42,75,24,8,35,13,11,23,9]*

print("\nArray : \n" ,nums ,"\n")

for a in nums:

    Temp\_Set.add(a)

Array = list(Temp\_Set)

size = len(Array)

products = []

pairs = []

for a in range(size):

*#Populating the Array with [(Number A)(Number B)(Their Product)]*

    for b in range(a + 1, size):

        product = Array[a]\*Array[b]

        products.append([Array[a],Array[b],product])

length = len(products)

*#Checking Each Element's Product in the Array and adding Number A and B to Array: Pairs if Equal*

for x in range(0,length -1):

*#store holds product of each element [(0:element A)(1:element B)(2:product)]*

    store = products[x][2]

    for y in range((x+1), length):

        comp = products[y][2]

*#print("A = " +str(products[x]) +" B: " +str(products[y]))*

        if(store == comp):

*#print(str(store) +" is Equal to " +str(comp))*

            pairA = [products[x][0]],[products[x][1]]

            pairB = [products[y][0]],[products[y][1]]

            pairs.append([[pairA],[pairB]])

*#Printing All The 2-Pairs*

if pairs:

    for z in range(0,(len(pairs))):

        print(str(pairs[z][0][0][0][0]) +" & " +str(pairs[z][0][0][1][0]) +" and " +str(pairs[z][1][0][0][0]) +" & " +str(pairs[z][1][0][1][0]) +" are a Two-Pair\n")

else: print("There are no Two-Pairs in this Array")

Question 5:

stack = []

**def** Filler(data):

    operator  = data.split()

    for op in operator:

        print (stack)

        if op in {"+", "-", "\*", "/"}:

            num2 = stack.pop()

            num1 = stack.pop()

            if op == "+":

                ans = num1 + num2

            if op == "-":

                ans = num1 - num2

            if op == "\*":

                ans = num1 \* num2

            if op == "/":

                ans = num1 / num2

            stack.append(ans)

        else:

            stack.append(int(op))

    return stack.pop()

*#Test Data:*

print(Filler("10 5 +"))

print(Filler("22 11 / 33 \*"))

*#print(Filler("100 2 + 30 b"))*

*#print(Filler("3 \* / 22 11 33"))*

Question 6:

import random as rd

from unicodedata import numeric

**def** Prime\_Checker(num):

    prime = True

*#Only Divisible by 1 and itself*

    for i in range(2,num):

        if ((num % i) == 0):

*#not a prime number*

            prime = False

            break

    if (prime == True):

        print(str(num) +" is Prime")

    else:

        print(str(num) +" is NOT Prime and can be divided:" ,i)

num = rd.randint(1,1000)

*#num = 12*

*#num = 3*

Prime\_Checker(num)

**def** SieveOfEratosthenes(n):

    prime = [True for i in range(n+1)]

    p = 2

    while (p \* p <= n):

*# If prime[p] is not changed, then it is a prime*

        if (prime[p] == True):

*# Update all multiples of p*

            for i in range(p \* p, n+1, p):

                prime[i] = False

        p += 1

*# Print all prime numbers*

    for p in range(2, n+1):

        if prime[p]:

            print(p)

*# Driver Code to Test different instances of n*

nums = [10,20,30,40,50,60]

i = rd.randint(0,(len(nums)-1))

n = nums[i]

print("Following are the prime numbers smaller than or equal to", n)

SieveOfEratosthenes(n)

Question 7:

'''

#Create a non balanced BST with inputs one at a time

#Needs Class, Object Node with left and right, methods for adding node to tree and possible visualisation using JSON

DIFFICULTY TO DISPLAY BST

'''

**class** Node:

**def** \_\_init\_\_(self, num):

        self.leftNode = None

        self.rightNode = None

        self.num = num

**def** addNode(root, node):

    if (root == None):

        node = root

    else:

        if (node.num < root.num):

*#place on left if not null*

            if(root.leftNode == None):

                root.leftNode = node

*#Recursion to Add on Left of Left Nodes below Root*

            else:

                addNode(root.leftNode, node)

*#else if(node.num > root.num)*

        else:

            if(root.rightNode == None):

                root.rightNode = node

*#Recursion to Add on Left of Left Nodes below Root*

            else:

                addNode(root.rightNode, node)

stop\_input = False

print("Enter Root Node: ")

root = Node(int(input()))

*#While loop to Ask User to input Node value*

while stop\_input == False:

    value = input("Enter A New Node Into The BST Or Press Enter To Exit: ")

*#Ignore Case for Exit value*

    if not value:

        print("Exiting Input Procedure...")

        stop\_input = True

    else:

        num = int(value)

        addNode(root, Node(num))

**def** bfs(node,level=0,res=[]):

  if level<len(res):

    if node:

      res[level].append(node.num)

    else:

      res[level].append(" ")

  else:

    if node:

      res.append([node.num])

    else:

      res.append([" "])

  if not node:

    return

  bfs(node.leftNode,level+1,res)

  bfs(node.rightNode,level+1,res)

  return res

**def** printBST(node):

  treeArray = bfs(node)

  h = len(treeArray)

  whiteSpaces = (2\*\*h)-1

**def** printSpaces(n):

    for i in range(n):

      print(" ",end="")

  for level in treeArray:

    whiteSpaces = whiteSpaces//2

    for i,x in enumerate(level):

      if i==0:

        printSpaces(whiteSpaces)

      print(x,end="")

      printSpaces(1+2\*whiteSpaces)

    print()

*#driver Code*

printBST(root)

'''

For Testing Purposes Part 1

print("Root: " ,root.num)

print("Left " ,root.leftNode.num)

print("Right" ,root.rightNode.num)

'''

'''

Part2

def display\_node(node):

    if (node.leftNode and node.rightNode):

        #for every level of depth -5

        print("{:>40}".format(node.leftNode.num),"{:>20}".format(node.rightNode.num))

        display\_node(node.leftNode)

        display\_node(node.rightNode)

    elif(node.leftNode):

        print("{:>40}".format(node.leftNode.num))

        display\_node(node.leftNode)

    elif(node.rightNode):

        print("{:>60}".format(node.rightNode.num))

        display\_node(node.rightNode)

print ("{:>48}".format(root.num))

display\_node(root)

'''

Question 8:

**def** newtonRaphson(n, iter):

    a = float(n) *# number to get square root of*

    for i in range(iter):

        n =(n + a / n)/2

    return n

*#number of iterataions (effects the accuracy of the final answer)*

iter = 1

"""

Testing Task 8

"""

print("With " ,iter ," iterations: \n")

print ("Approximate Square Root for 9: " ,newtonRaphson(9,iter) ,"\n")

print ("Approximate Square Root for 2: " ,newtonRaphson(2,iter) ,"\n")

print ("Approximate Square Root for 12: " ,newtonRaphson(12,iter) ,"\n")

print ("Approximate Square Root for 37: " ,newtonRaphson(37,iter) ,"\n")

print ("Approximate Square Root for 53: " ,newtonRaphson(53,iter) ,"\n")

print ("Approximate Square Root for 9: " ,newtonRaphson(25,iter) ,"\n")

Question 9:

array = [1,2,3,13,5,6,51,7,30,8,12,9,10,11,1,2,3,4,6,13,7,8,9,11,50,51]

*#repeated should be 1, 2, 3, 6, 7, 8, 9, 11, 13, 51*

'''

array = [1,2,3,4,5,6,7,8,9,10,11,24,67,41,72]

should have no repeated elements

'''

print("Array: \n" ,array)

repeated = set()

not\_repeated = set()

for x in array:

    current\_length = len(not\_repeated)

    not\_repeated.add(x)

*#After Adding Each element to set, I am checking if length grew to detect if set accepted element or if it is repeated*

    if(len(not\_repeated) != current\_length + 1):

        repeated.add(x)

*#Checking if length of non repeated elements is the same as original array*

if(len(not\_repeated) == len(array)):

    print("\nThere are no repeated elements in this Array")

else:

    print("\nThe repeated values in the Array are:\n" ,list(repeated))

Question 10:

*#nums = [1,2,55,3,13,5,6,30,8,14]*

*#nums = [5,5,5,5,5,5]*

nums = [0,-2,-3,-1,-55,-9]

print(nums ,":\n")

**def** find\_largest(list):

*#largest element is the only element*

    if len(list) == 1:

        return list[0]

    else:

        current\_largest = find\_largest(list[1:])

        if current\_largest > list[0]:

            return current\_largest

        else:

            return list[0]

print("Largest Number in List: " ,find\_largest(nums))

Question 11:

import math

**def** cosine(x, n):

    cos = 0

    for n in range(n):

        cos += ((-1)\*\*n) \* (x \*\* (2\*n)) / (math.factorial(2 \* n))

*# num \*\* x means num^x*

    return cos

**def** sine(x,n):

    sin = 0

    for n in range(n):

        sin += ((-1)\*\*n) / math.factorial(2\*n+1)\*(x\*\*(2\*n+1))

    return sin

print("Enter the order (n) for the expansion: ")

n = int(input())

*#n = 1*

print("Enter the value of x:")

x = float(input())

print("Cos(",x ,") to order",n ,"=",cosine(x,n),"\n")

print("Sin(",x ,") to order",n ,"=",sine(x,n))

Question 12:

fibonacci = [0,1]

sum = 0

print("Enter the value for n: ")

n = int(input())

error = "!! n cannot be equal to or less than 0 !!"

**def** Fibonacchi\_Sequence(fibonacci,n,sum):

    for i in range(2,n):

        next = fibonacci[-1] + fibonacci[-2]

        fibonacci.append(next)

    print("The Fibonacci Sequence up to the first",n ,"elements is : \n" ,fibonacci ,"\n")

    for i in range(0,n):

        sum += fibonacci[i]

    print("The Fibonacci Sequence SUM of the first",n ,"elements is:",sum,"\n")

if(n <= 0):

    print(error)

elif(n == 1):

    fibonacci.remove(1)

    Fibonacchi\_Sequence(fibonacci,n,sum)

else:

    Fibonacchi\_Sequence(fibonacci,n,sum)