ICT1054

Data Structures and Algorithms

Icon

Description automatically generated

University of Malta

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

Luigi Naudi

0249603(L)

Contents:

Statement of Completion.............................................................................................................. 2

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

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

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

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

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

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

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

Question 6.....................................................................................................................................0

Question 7.....................................................................................................................................0

Question 8.....................................................................................................................................0

Question 9.....................................................................................................................................0

Question 10...................................................................................................................................0

Question 11...................................................................................................................................0

Question 12...................................................................................................................................0

Text, letter

Description automatically generated



Statement of Completion:

Question 1 - Completed and Works

Question 2 – Completed and Works

Question 3 – Completed and Works

Question 4 – Attempted and ran into a few bugs; now fixed and Works Well:

* Duplicate Values were not filtered inside the array
* Duplicate two-pairs were printing multiple times

Question 5 – Completed but has some bugs:

* Cannot handle non-RPN expression
* Crashes when coming across invalid char

Question 6 – Completed and Works

Question 7 – Attempted but has missing feature:

* Cannot display BST

Question 8 – Attempted and Works

Question 9 – Completed and Works

Question 10 – Attempted and Works

Question 11 – Not Yet Attempted

Question 12 – Not Yet Attempted

**Question 1**

References: []

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**

References: []

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**

References: []

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**

References: []

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**

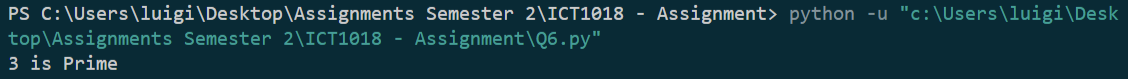
References: []

**Question 6**

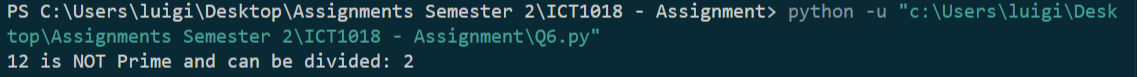
References: []

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.

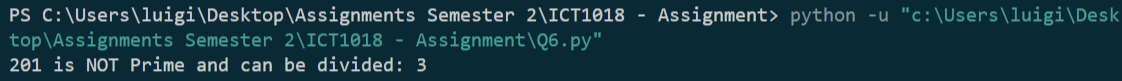
Prime Number Test:



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: [x]

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 [x] and works well. It was tested using a range of outputs and produced results as expected:

Text

Description automatically generated

Text

Description automatically generated

<https://stackoverflow.com/questions/34012886/print-binary-tree-level-by-level-in-python>

**Question 8**

References: []

**Question 9**

References: []

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**

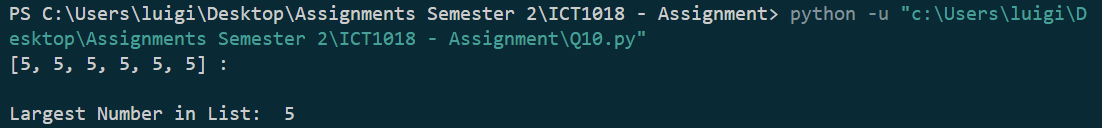
References: []

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: []

**Question 12**

References: []