COMP 6481: ASSIGNMENT 1
Dept. of Computer Science & Software Eng., Concordia University
COMP 6481 --- Fall 2022
Programming and Problem Solving

Himangshu Shekhar BARUAH, ID 40229774

Part I

Question 1

a) Given an array of integers of any size, n≥1, and a number m, develop an algorithm as a pseudo code (not a program!) that would swap all numbers with value m with the subsequent numbers. You must perform this operation in place i.e., without creating an additional array and keeping the number of operations as small as possible. For example, given an array [5, 4, 6, 9, 6, 3, 6], and a number 6, the algorithm will return [5, 4, 9, 6, 3, 6, 6]. Finally, your algorithm must not use any auxiliary/additional storage to perform what is needed.

Ans. The pseudo code for the above problem is as follows:

```
FOR: i ← 0 to n-1
        IF: array [i] is equal to m
                array[i] = sum of array[i] and array[i+1]
                array[i+1] = difference of array[i] and array[i+1]
                array[i] = difference of array[i] and array[i+1]
                increment index by 1
        END IF
END FOR
```

b) What is the time complexity of your algorithm, in terms of Big-O?

Ans. The time complexity is O(n) as there is one for loop which runs till n-1 times.

c) What is the space complexity of your algorithm, in terms of Big-O?

Ans. The space complexity is O(1) as the requirement of space is O(1) per call in the loop.

Question 2

a) Given a string of random length and random contents of characters, that do not include special characters, write an algorithm, using pseudo code that will rearrange the string with all its consonants followed by all its repeating characters followed by all its vowels. For instance, given "assignment1" the algorithm must return: "gmtsn1aie".

Ans. The pseudo code for the above problem is as follows:

```
FOR: i ← 0 to length of input
        FOR: j ← i + 1 to length of input
                IF: character at input[i] is equal to character at input[j]
                        Add to list duplicate
                END IF
        END FOR
END FOR

FOR: i← 0 to length of input
        IF: Arraylist duplicate does not contain input[i]
                IF: character at input[i] is equal to either of a,e,i,o,u
                        Add to list vowel
                ELSE IF: character at input[i] is equal to either of 1,2,3,4,5,6,7,8,9,0
                        Add to list numbers
                ELSE:
                        Add to list consonants
                END IF
        END IF
END FOR
RETURN: In the order → list consonants, list duplicates, list numbers, list vowels
```

b) What is the time complexity of your algorithm, in terms of Big-O?

Ans. The time complexity of the algorithm is $O(n^2)+O(n)$, which can be reduced to $O(n^2)$.

c) What is the space complexity of your algorithm, in terms of Big-O?

Ans. The space complexity of the algorithm is O(n) as there may be n elements to be added in the lists.

Question 3

i) Develop a well-documented pseudo code that finds two farthest tetradic number (a four-way number) elements in an array with their difference being 10 and two successive tetradic number elements with the largest difference. The code must display values and indices of these elements. For instance, given the following array [11, 111, 88, 101, 181, 808, 1, 818, 0] your code should find and display something like the following (notice that this is just an example. Your solution must not refer to this example.

Two farthest tetradic elements with their difference equal to 10 are 11 and 1 which have 5 elements between them. Two consecutive tetradic elements with the largest difference are 818 and 0. In case of multiple occurrences of pairs with a given difference or largest difference, your code must display the first found pair.

Ans: The pseudo code for the above is as follows:

```
Algorithm farthestTetradic( Arr, diff )
        Input array Arr containing tetradic numbers
        Output farthest elements with difference diff

        for i ← 0 to Arr length do
                if arr[i] = reverse arr[i] and arr[i] contains only (1,8,0) then
                        Add to list tetradic ← arr[i]

        for i ← 0 to tetradic length do
                for j ← i+1 to tetradic length do
                        if  | tetratic[i] - tetradic[j] | = diff then
                                save indexes to arrayFarthest
        Return  values to indexes of arrayFarthest

Algorithm largestDifference( Arr, diff )
        Input array Arr containing tetradic numbers
        Output largestDifference

        max ← 0
        for i ← 0 to Arr length do
                if arr[i] = reverse arr[i] and arr[i] contains only (1,8,0) then
                        Add to list tetradic ← arr[i]

        for i ← 0 to tetradic length - 1 do
                if  | tetratic[i] - tetradic[i+1] | >  max then
                        save indexes to arrayFarthest
        Return  values to indexes of arrayFarthest
```

ii) Briefly justify the motive(s) behind your design.

Ans. First the tetradic numbers are stored in a list from the main array by checking if they are palindrome and contain 0,1,8 digits only. Then, two for loops are used to compare the difference between the pair of numbers that gives the difference given and the numbers that are returned. Also for the largest consecutive difference the tetradic number array is iterated once and compared with the next element and the max difference is stored and the numbers returned.

iii) What is the time complexity of your solution? You must specify such complexity using the Big-O notation. Explain clearly how you obtained such complexity.

Ans. For finding the farthest numbers, two for loops are used so the time complexity is O(n^2).
For finding the largest difference, only one loop is used so it has a time complexity of O(n).

iv) What is the maximum size of stack growth of your algorithm? Explain clearly

Ans. The maximum size stack growth will be O(n) as worst case n tetradic numbers need to be stored in the array.