Name: - Olohi Goodness John
Peers: - Sanjana Yasna
References: -

Part 1: Algorithmic Analysis

For each of the questions (1-3) below, answer the following:
What are the following algorithms (written in pseudo-code) computing?
Describe, in English, their input, output, and how the output is related to the input.
Analyze the complexity (in terms of running time only) of the algorithm.

Question 1

```
mysteryFunction1 (array a of size n):
  x=0
  for i from 0 to n - 1:
    x = x + a[i]
  return x
end
```

- Description: The algorithm, presented above, iteratively computes the sum of values in the array, a.
- Input and Output: The function takes in an array, a, of size n, and returns the sum of elements in the array. Since the type of elements in the input array, a, is not specified, we cannot conclude on what the return type should be, e.g., a string concatenation vs an integer sum.
- The runtime of the algorithm is $O(n)$, where n represents the length of the array

Question 2

```
mysteryFunction2 (array a of size n):
  if n = 1
    return a[0]
  else
    x = a[n-1] + mysteryFunction2(a[0...(n - 2)])
    // Note: a[0...(n - 2)] = the first n - 1 elements of array a
  return x
end
```

- Description: The algorithm, presented above, recursively computes the sum of values in the array, a.
- Input and Output: The function, takes in an array, a of size n, and returns the sum of elements in the array by recursion. If we assume the contents of the array are ints, then the function returns an integer, x, representing the sum of values in a. If we assume the contents of the array are strs or chars, then the function returns a string, x, which is a concatenation of the elements in a
- The runtime of the algorithm is also $O(n)$ where n represents the length of the array

Question 3

```
mysteryFunction3 (array a, integer x):
  l = 0
  m = 0
  n = len(a) - 1
  while l <= n:
    m = (l + n) / 2
    if x == a[m]
      return m
    else if a[l] <= a[m]:
      if x > a[m]:
        l = m+1
      else if x >= a[l]:
        n = m - 1
      else:
        l = m + 1;
    else if x < a[m]:
      n=m-1
    else if x <= a[n]]:
      l=m+1
    else:
      n = m - 1
  return -1;
end
```

- Description: The algorithm, presented above, returns the index of an integer, x, in a rotated sorted array or -1, if the integer is not present in the array.

- Input and Output: The function, takes in an array, a of size n and an integer, x. It searches for x within the array and returns its index. If not found (or in some cases,if the array is not a sorted rotation), it returns -1. The algorithm is designed to search correctly for rotated arrays.
- The runtime of the algorithm is $O(logn)$ where n represents the length of the array