Mock Exam - Part 1

All complexity questions should be expressed in big O notation.

1. In your words, explain the difference between an algorithm and a data structure.

2. Giving the following code

```
function f(n)
    if n=1
        return 1
    else
        return f(n-1) * f(n-2)
```

What is its complexity?

3. What is the complexity of the recursive version of the factorial (n) function

4. What is the complexity of an if-then statement?

5. Giving the following nested loops

```
for i = 0 to i < n
    for j = 0 to j < n * n
        for k = 0 to k < j
```

What is its complexity?

6. Explain in your own words the difference between Dynamic programming and Brute force?
7. Explain in your own words the difference between Dynamic programming and Backtracking?
8. Run at the following code and explain what it does. You may want to change the lists in the variable arr to try to understand the results.

```
M = 4

def maximumSum(a, n):

  prev = max(max(a))

  Sum = prev

  for i in range(n - 2, -1, -1):

    max_smaller = -10**9

    for j in range(M - 1, -1, -1):

      if (a[i][j] < prev and  a[i][j] > max_smaller):
```

```
                max_smaller = a[i][j]
            if (max_smaller == -10**9):
                    return 0
            prev = max_smaller
            Sum += max_smaller
    return Sum


arr = [[1, 7, 3, 4],
      [4, 2, 5, 1],
      [9, 5, 1, 8]]
n = len(arr)
print(maximumSum(arr, n))
#End
```

For example, you could char arr for arr[][] = {{9, 8, 7}, {6, 5, 4}, {3, 2, 1}}

9. What is the complexity of the code from point 8?
10. The code from 8, is following a brute-force, a recursive or a greedy strategy? Briefly explain your answer.