Important Information

This set of solutions should NOT be the first thing you look at, We encourage you only to look at these after you have considered the guidance. You will learn very little from memorising the solutions to these and the exam questions will be different. You will also learn far more by discussing your answers with another person than you will from reading our solutions. These solutions are not the best or only solution, please contact your lecturer if you are uncertain (as your solution may still be correct even if it does not match the given solution)
All Python code you write for this exam must satisfy the following requirements:

- Syntax should satisfy Python 3 requirements
- Use syntax, modules, structures and constructs presented in lectures.
- Avoid using in built libraries that are performing non-trivial operations
- to have the most realistic exam experience, you should avoid actually running any of the code given (or that you produce) until after you have finished everything

Write down any assumptions you make.
The table below is included for your convenience

Question	Points	Score
1	1	
2	1	
3	1	
4	1	
5	7	
6	5	
7	9	
8	6	
9	8	
10	8	
11	7	
Total:	54	

0

Question 1: [1 marks]

what will the following code output?

```
def doThing(n):
        S = 0
        for i in range(n,0,-3):
                S+=i
        print(S)

doThing(9)
```

A. 9

B. 18

C. 27

D. the code will not run or will run indefinitely

Question 2: [1 marks]

what will the following code output?

```
def doSomethingElse(x):
        if x > 0:
                x = -x
        while x < 0:
                if abs(x)>1:
                        x *= abs(x)
                else:
                        x += 1
        return x

print(doSomethingElse(10))
```

A. 10

B. -10

C. -100

D. 0

E. the code will not run or will run indefinitely

2

Question 3: [1 marks]
Which of the following principles does merge sort use to sort a list?

A. greedy

B. backtracking

C. divide and conquer

D. brute force

Question 4: [1 marks]
What is the relationship between a heap and heapsort?

A. heapsort is done by successively extracting the minimum from a heap

B. heapsort is the act of successively inserting items into a heap; this heap is then a sorted list

C. a heap can only be constructed via heapsort

D. there is no relationship, their naming is coincidental

2

Question 5: [7 marks]

Imagine a problem of finding a subset of a graph where each vertex in this subset has a different number of edges. Let's call this problem the *uniqueDegree problem.* You are given the decision problem variant of this where a set must be found of size k

(a) (1 mark) Describe what the certificate to this problem might include?

a certificate here is a set of vertices which we believe to each have a different degree and the number of vertices in this set should be k

(b) (4 marks) Provide a python function which (given a graph in adjacency Matrix format, a certificate to this problem, and a value of k) can determine whether this certificate is a solution to the problem for the given value of k.

- checks the certificate has k vertices (1)
- considers every vertex in the certificate (1)
- counts the number of edges of that vertex (1)
- compares that degree against degrees of those previously determined (1)

(c) (2 marks) Based on your previous function; in what class of problem is the *uniqueDegree* problem? Explain your answer.

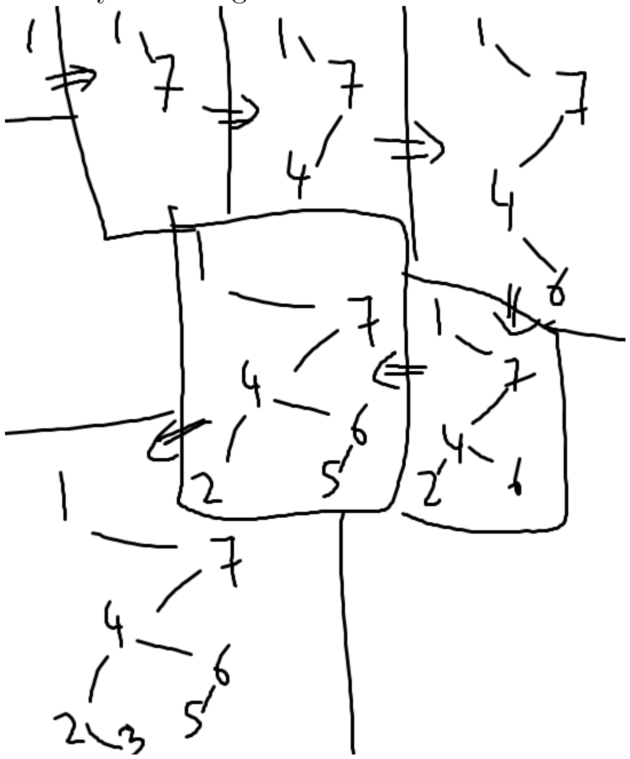
note that your function would likely be a P algorithm, it's verifying a certificate to the problem in P time therefore this is at worst an NP problem (if you can come up with a P algorithm to solve it (which I doubt) then you could argue it is a P problem)

7

Question 6: [5 marks]

This question is about Binary Search Trees:

(a) (2 marks) given the following list of items `[1,7,4,6,2,5,3]`, create a Binary Search Tree by inserting each into the tree in the order they appear in the list given



(b) (1 mark) give a new ordering of these values in the list such that when inserted into a binary search tree in the order they appear, the BST is balanced

one possible ordering might be `[4, 2, 6, 1, 3, 5, 7]`

(c) (2 marks) Explain what impact the balance of a binary search tree has on the complexity of its find operation and why.

an highly unbalanced tree will have a height of N where n is the number of nodes in the tree whereas a balanced tree has height of logN. Given that find requires you traverse a single path to the end, the height of the tree determines the length of individual paths. Hence a less balanced tree will take longer to run a find operation on

5

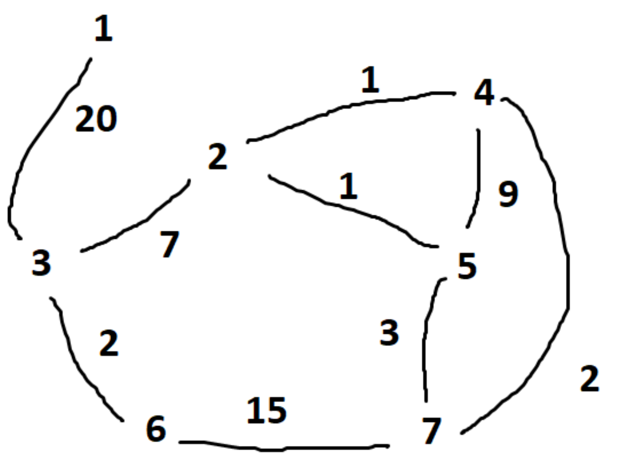
Question 7: [9 marks]

Consider the undirected graph G which includes the following edges:

Table 1: edges in graph

start vertex	end vertex	cost
1	3	20
2	3	7
2	4	1
2	5	1
3	6	2
4	5	9
4	7	2
5	7	3
6	7	15

(a) (2 marks) draw the graph that corresponds to this set of edges (including the weights)



(b) (2 marks) produce an adjacency list that represents this graph; please put weights in brackets

the adjList would look like the following:

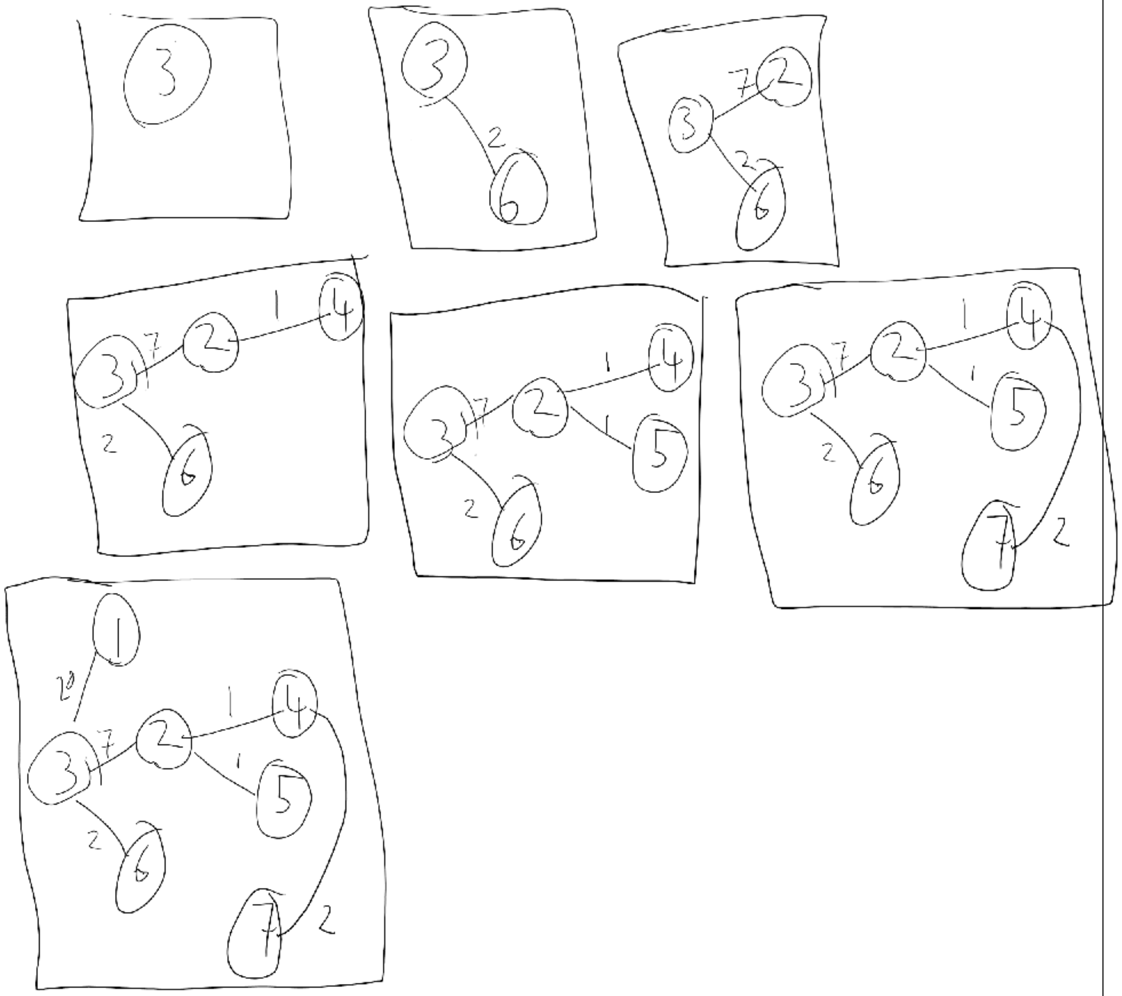
```
1 -> 3(20)
2 -> 3(7),4(1),5(1)
3 -> 1(20),2(7),6(2)
```

4

```
4 -> 2(1),5(9),7(2)
5 -> 2(1),4(9),7(3)
6 -> 3(2),7(15)
7 -> 4(2), 5(3), 6(15)
```

0

(c) (4 marks) Using prim's algorithm, find the minimal spanning tree of this graph. Make sure you show the result after selecting each edge; you should start with vertex 3.



(d) (1 mark) Suggest a useful invariant of Prim's algorithm for the kth iteration of the outer loop.

at the kth iteration of Prim's algorithm we will have included k+1 vertices in the spanning tree (or equally k edges) and the selected edges will not include any cycles

5

Question 8: [6 marks]

consider the NQueens problem for a N of 4

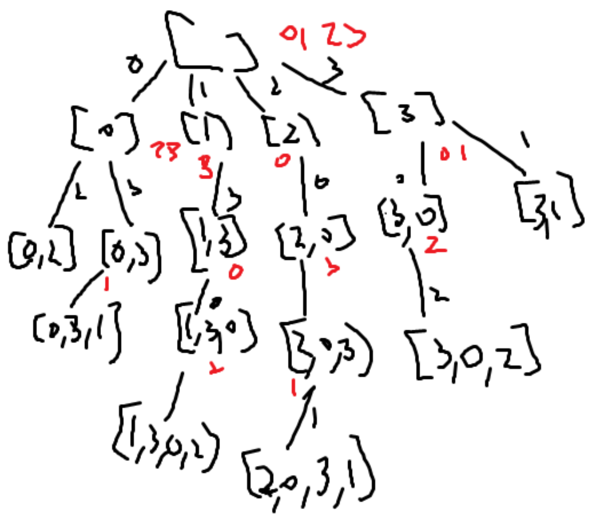
(a) (1 mark) what form would partial solutions take?

a list of Queen positions such as [1,3] where we have tried to place a queen in column 0 row 1 and column 1 row 3

(b) (1 mark) How would one be able to tell that a partial solution is in fact a complete solution?

the partial solution would not have any possible attacks in it and it would have N items in it.

(c) (4 marks) Draw a backtracking tree for this problem using the single list representation for N Queens



6

Question 9: [8 marks]

This question is about sorting

(a) (2 marks) Explain how insertion sort sorts a list

students should mention

- there is a sorted section on the left (or right)
- new items are incrementally included into the sorted section
- this is done by shifting the new element back until the sorted section is still sorted

(b) (2 marks) Give an example of list of numbers for which insertion sort would take the most time, and an example of a list of numbers for which insertion sort would take the least time. Make sure both lists are of the same size and identify which corresponds to which situation.
you may assume this version of insertion sort produces something in ascending order

we are expecting one situation where the list is in sorted order and one where the list is in reverse sorted order.
For example [1,2,3,4] would take the least time as nothing needs to move in the list
alternatively [4,3,2,1] would take the most time as every item must be shifted as far to the left as possible with each iteration.

(c) (1 mark) Given the following list of numbers, show what quicksort would partition this list into:
`[10,6,2,73,2,8,7]`
you may assume a pivot of 7

```
left: [6,2,2]; pivot 7; right [10,73,8]
```

5

(d) (1 mark) What is the base case for quick sort?

the ideal base case is a list of size 1 as there is not partitioning to do.

(e) (2 marks) how does quicksort use the results of the partition function to continue sorting the list?

- the new position of the pivot tells quick sort where to split
- quicksort is then run of the end to the left of the pivot and the right of the pivot
- after this the results of running quick on the left is joined to the pivot and the results of running it on the right
- this gives a sorted list which is sent upwards to the previous call of quick-sort

3

Question 10: [8 marks]

(a) (4 marks) Write a python function to **recursively** find the minimum value in a list; this should accept a list as an argument and return the index of the smallest item.
For example given the list as follows: `[9,-2,6,1,80,9,-2]`, your function should return either 1 or 6 (which are the indices of -2).

this would possibly be:

```
def getMin(aList):
        return getMinAux(aList, 0)

def getMinAux(aList, pos):
        if (len(aList) - 1) == pos:
                return pos
        else:
                rest = getMinAux(aList, pos + 1)
                if aList[rest] < aList[pos]:
                        return rest
                else:
                        return pos
```

- correct base case (1)
- correct recursive call (1)
- correctly combines with results of later calls (1)
- correct function definition (0.5)
- correctly returns results (0.5)

(b) (4 marks) Write a python function that uses your previous function to make a list of the k smallest items in a list.
For example, given the list `[9,-2,6,1,80,9,-2]` and a k of 4, your function should print (or return) `[-2,-2,1,6]`

- correctly runs their getMin function from before (1)
- this is run k times (1)
- each time it ignores the minimum values already obtained (1)
- correct function definition (0.5)

8

- correctly returns results (0.5)

0

Question 11: [7 marks]

(a) (1 mark) give an example of an algorithm.

almost anything could go here like driving a car, peeling a banana, sending a letter, etc.

(b) (5 marks) explain why your example is an algorithm.

students should address the following aspects of algorithms in their answer:

- finiteness
- sequence of steps
- effectiveness
- definiteness
- input (zero or more)
- output (1 or more)

taking the peeling a banana as the example, it demonstrates a sequence of steps as you can provide a set of steps to follow in order (ex. break the tip, pull down the skin to the base of the banana, rotate and pull down the skin again to the base; repeat this process of turning and peeling until all of the skin has been peeled.)
this demonstrates finiteness as there is a clear termination point, eventually the banana would be peeled by these instructions as it has a finite amount of skin to peel off
it is effective and definite as there is no ambiguity about how to do these steps and each step is small enough to be acheivable (although perhaps in my instructions breaking the tip should suggest a direction to break it to)
input here is an unpeeled banana
output is a peeled banana
as each of these has been met it is an algorithm

(c) (1 mark) how does an algorithm differ from a process?

the only difference is on finiteness, the student's response should make this clear. See previous question for the features of an algorithm

7