Algorithms and Datastructures

Assignment 5

Lucas van der Laan
s1047485

1

(a) The total cost is 1 + 1 + 2 + 3 + 3 + 4 + 5 = 19

(b) A total of 12
For every starting point at least 1, so that is 8.
For every starting node that has n (>1) neighbouring edges with the same weight, this happens with E, B, C, G, thus we have an extra 4 MSTs
Examples:
A $\rightarrow$ E $\rightarrow$ F $\rightarrow$ B $\rightarrow$ G $\rightarrow$ H $\rightarrow$ C $\rightarrow$ D
E $\rightarrow$ A $\rightarrow$ F $\rightarrow$ B $\rightarrow$ G $\rightarrow$ H $\rightarrow$ C $\rightarrow$ D
E $\rightarrow$ F $\rightarrow$ A $\rightarrow$ B $\rightarrow$ G $\rightarrow$ H $\rightarrow$ C $\rightarrow$ D

(c)
1. [(E,F)], added (E, F)
2. [(E,F), (A, E)], added (A, E)
3. [(E,F), (A, E), (B, E)], added (B, E)
4. [(E,F), (A, E), (B, E), (F, G)], added (F, G)
5. [(E,F), (A, E), (B, E), (F, G), (G, H)], added (G, H)
6. [(E,F), (A, E), (B, E), (F, G), (G, H), (G, C)], added (G, C)
7. [(E,F), (A, E), (B, E), (F, G), (G, H), (G, C), (G, D)], added (G, D)

2

Algorithm 1 Least amount of cell phones on a road

```
Require:
    xs - The list of houses
    Q - xs as a queue
ys ← []
savedDistance ← Pop(Q)
while not Empty(Q) do
    distance ← Pop(Q)
    if distance - savedDistance $>$ 8 then
        Append(ys, savedDistance + 4)
        savedDistance ← distance
    end if
end while
if savedDistance - Last(ys) $>$ 4 then
    Append(ys, savedDistance)
end if
```

Example: 1,7,10,15,16,17,19,40,44,45,50
7 - 1 = 6 > 8 –> Nothing
10 - 1 = 9 > 8 –> ys.append(5)
15 - 10 = 5 > 8 –> Nothing
16 - 10 = 6 > 8 –> Nothing
17 - 10 = 7 > 8 –> Nothing
19 - 10 = 9 > 8 –> ys.append(14)
40 - 19 = 21 > 8 –> ys.append(23)
44 - 40 = 4 > 8 –> Nothing
45 - 40 = 5 > 8 –> Nothing
50 - 40 = 10 > 8 –> ys.append(44)
50 - 44 = 6 > 4 –> ys.append(50)

3

We first make a graph of all the people with their connections, every node exists of the identifier of the person and the how many edges are connected. We then go over every node and check if the total number of edges is bigger or equal to 5 and that the total amount of people left minus the amount of edges that the node has is bigger or equal to 5. This satisfies both constraints of knowing at least 5 people and not knowing 5 people. We do this until no changes have been made in the checking and then we can return the list of people that satisfy that by going over every node.
The running time is n^3.
For correctness: We know it terminates, because if it keeps removing people then the list will be empty and that gets returned, and if the doesn't then it terminates. We know it gets the correct people, because at the end of the last iteration the only people that are left are people that know at least 5 people and don't know at least 5 people.

4

1. False, as the edge can be connected to a node that only has a single edge, thus it has to get added.

2. False, because the shortest path does not need to contain every edge between nodes that the MST has. This can be seen in a triangle of [((A, B), 2), (A, C), 2), (B, C), 1)], dijkstra's shortest path will instantly go from A $\rightarrow$ B and ignore C, while the MST requires C to also be found.

3. False, every smallest edge (that is not connected to an already discovered node) has to be in there, otherwise it cannot be an MST.