1 Part 1 Theory

1.1 Task 1 The interval scheduling problem

a) How interval scheduling returns an optimal set

This algorithm always choose an event that ends as early as possible. The algorithms will always choose the event that finished as early as possible for all elements of R. When the algorithm does this, it will always returns an optimal set.

b) Interval scheduling a table of activities

We have the following table of activities:

Activity	U	V	W	X	Y	Z
Start time	3	2	5	5	10	1
Finishing time	6	4	8	9	12	5

After running the interval scheduling algorithm, it will choose the V element first as it has the smallest finishing time. After that, it will move on to the W element and lastly the Y element. The size of A will then be $A = [V, W, Y]$.

1.2 Task 2 Minimum spanning trees

a) Edge (u,v) with lower weight than all other edges in a connected graph

The minimum spanning tree is a subset of the edges in a connected undirected graph. This is the spanning tree whose sum of edge weights is the lowest possible weight. If we therefore have edge (u,v) that has a lower weight than all other edges in the graph, it will therefore be included in the minimum spanning tree.

b) Consider a statement about Minimum Spanning Trees

We have a weighted undirected graph $G = (V,E)$ where all of the edges are weighted differently. The nodes V are divided into two disjoint sets X and Y. We consider Kruskal's algorithm to find the minimum spanning tree. This algorithm maintains a forest, initially consisting of unconnected individual vertices and disjointed sets of data. To compute the minimum spanning tree, the algorithm will find the lowest weighted union between the disjointed data sets, meaning the minimum spanning tree will include the edge between X and Y with the lowest weight.

1.3 Task 3 Shortest path in a graph

a) Djikstra's algorithm to solve the shortest path problem on an undirected graph

The shortest path problem involves finding the shortest path between two nodes in a graph. The shortest path such that the sum of weights of the graphs edges is the lowest possible. Djikstra's algorithm finds the shortest path between between nodes. It starts at a given source node and picks unvisited nodes with the lowest distance, and calculates the distance through it to each unvisited neighbour and then updates the neighbours distance if it is smaller.

b) Why Djikstra's algorithm work

Djikstra's algorithm works because all edge weights are non-negative values, and the node with the lowest weight is always chosen. Because the path with the lowest weight is always chosen, it will always compute the shortest path.