Assignment 1

Assignment 1 is divided into 2 parts:

- **Part 1:** Knapsack Problem **(Maximum of 3 points)**
  - Pre-requisites: None
- **Part 2:** Shortest path from A to B **(Maximum of 2 points)**
  - Pre-requisites: Minimum of 2 points in knapsack problem

PART 1

The Knapsack problem (BKP) is a problem in combinatorial optimization. This problem is described as follows:

Given a list of items, each item has a weight and a benefit, determine the number of each item to include in a bag so that the total weight is less or equal to the limit of the bag and we get the maximum benefit.

The most common problem to solve is the 0-1 knapsack problem, which restricts the number of copies of an item to one. Therefore, an item can be inside the bag or outside.

We now consider the problem with a set of items (you can find the document in canvas with the name "assignment 1 knapsack.txt").These items are represented by 2 values (weight and benefit) as illustrated in the table below.

Item ID	Benefit (b)	Weight (w)
1	20	15
2	40	32
…	…	…

What to do: Your assignment now is to apply Breadth-first search (BFS) and Depth-first search (DFS) to search for the best combination of items inside the bag. **Remember, only one copy of an item.** You need to use a tree (queue or stack depending of the algorithm) and nodes in order to implement both search strategies. You need to present both codes to the teacher. **(0.5 points each code).**

Pre-conditions in order to present your code:

- The codes should run faster than 0.5 seconds. If you cannot get it faster, talk to the teacher. The reason could be your computer.
- You must use queue or stack, depending of the algorithm (or simulate them using an array).

Your report has to cover the key parts as follows:

1. - Explanation of the problem. **(1 point)**
   a. Give the representation of a solution (answer) of the problem, as explained during the course**. (0.5)**
   b. Give the equation of the objective function (what we want to maximize). **(0.25)**
   c. Give the equation for the restriction(s) of the problem. **(0.25)**
2. Comparison of the algorithms. **(1 point)**
   a. Comparison of the time expended by the algorithms. **(0.5)**
   b. Comparison of the space used in memory at a time by the algorithms. **(0.5)**

PART 2

Find the shortest path in a graph from one node to another is one of the most important problems in AI.

During the lectures, we were practicing a lot with a graph of Rumania's main cities and roads. The goal of this problem was to find a path from ARAD to BUCHAREST, sometimes we were finding the best path sometimes we were fine with one path between the two cities.

In this Assignment, we will have the map of Spain's main cities and road from city to city. The initial city will be MALAGA and the goal city will be VALLADOLID. In the document (Assignment 1 Spain map), you will find the following information:

City A City B Cost (You can go from city A to city B and from city B to city A)

City A City C Cost

….

City A Straight_Line_Distance_to_goal

City B Straight_Line_Distance_to_goal

….

What to do: Your assignment now is to apply Greedy Best-first search and A*, implementing them as it was explained in the lectures, to find the path from **MALAGA** to **VALLADOLID** (Greedy Best-first search will return the first solution found and A* will return the optimum path). You need to present both codes to the teacher. **(0.25 points each code).**

Your report has to cover the key parts as follows:

1. – Explanation of the problem **(1.5 points)**
    a. Give the representation of a solution (answer) of the problem, as explained during the course. **(0.5)**
    b. Give the equation of **f(n)** used in Gready Best-first Search (or Explain how to calculate f(n)). **(0.25)**
    c. Give the equation of **f(n)** used in A* (or Explain how to calculate f(n)). **(0.25)**
    d. Explain both algorithms and the differences between them. **(0.5)**

Conditions to approve the assignment 1

In order to receive the points for the code, you need to submit the report and have more than half of the points. Example: If a report has 1 point, then you need to have a minimum of 0.5.

- The score to pass part 1: 2 points.
- The score to pass part 2: 1 point.
- Minimum score to pass assignment 1: 3 points.

SCORES 3-5

- Your score < 3: U
- 3 <= your score <= 3.75: 3
- 3.75 < your score < 4.75: 4
- 4.75 <= your score: 5

SCORES A-F

- Your score < 3: F
- 3 <= your score < 3.4: E
- 3.4 <= your score < 3.8: D
- 3.8 <= your score < 4.2: C
- 4.2 <= your score < 4.6: B
- 4.6 <= your score: A

SUBMISSION DETAILS

Maximum Score: 5 points.

I will not consider a submission without the following format:

File names for the submission:

StudentName_StudentLastName_Assignment1_Part1_Code.zip (Only by email)

StudentName_StudentLastName_Assignment1_Part1_Report.pdf (Only in canvas)

StudentName_StudentLastName_Assignment1_Part2_Code.zip (Only by email)

StudentName_StudentLastName_Assignment1_Part2_Report.pfd (Only in canvas)

Email subject for the submission:

[DVA340] StudentName StudentLastName Assignment1 Code

ADDITIONAL INFORMATION!

You are allowed to send each part of the assignment a maximum of three times. When you approve the assignment, you are not allowed to submit the assignment again.

Before defending the code to Miguel Leon, you have to send the code by email to johan.hjorth@mdu.se

Before submitting the report, you should present this assignment first to Johan Hjorth. After that, and only if everything is correct, you are able to submit the report in canvas.

You cannot get the points of the report without presenting the code first.