The Knapsack Problem (BKP) is a crucial challenge within combinatorial optimization. To solve complex search problems, it introduces weight and benefits to maximize utility within some given constraints.

Objective:

This task presents you with items, each represented by distinct weights and benefits. The task is to find items to fill a knapsack, ensuring the total weight does not exceed the knapsack's capacity while maximizing the overall benefit. This assignment explicitly addresses the 0-1 Knapsack Problem, where each item can be included once or excluded altogether.

Item representation:

The items for consideration are detailed in the document "assignment 1 knapsack.txt Download assignment 1 knapsack.txt" and are accessible through Canvas. Below is a representation of items and details:

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

Task Overview:

Your task is to implement Breadth-first Search (BFS) and Depth-first Search (DFS) methodologies to find the optimal combination of items. These strategies necessitate conceptualizing a tree, navigated with a queue (BFS) or a stack (DFS) to explore potential item combinations. After implementing these two search strategies, you are supposed to present them to a lab assistant.

Part 2

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

During the lectures, we've been looking at a graph of Rumania's main cities and roads. The goal presented in the lectures was to find a path from ARAD to BUCHAREST, sometimes we found the best path, but sometimes we were fine with any path between the two cities.

In this assignment, we will instead look at a map of Spain's main cities and roads between them. The initial city in this new map 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
City A	City C	Cost
...		
City A	Straight_Line_Distance_to_goal	

City B	Staright_Line_Distance_to_goal	
...		

Objective

Your assignment now is to apply Greedy Best-first search and A*, implementing them as it was explained in the lectures. Greedy Best-first search will return the first solution found and A* will return the optimum path. You need to present both your codes to a teacher.

Your report has to cover the key parts as follows:

Part 1 - knapsack problem:

1.
    - Explanation of the problem. (2 points)
        - Give the representation of a solution (answer) of the problem, as explained during the course. (0.5)
        - Give the equation of the objective function (what we want to maximize). (0.5)
        - Give the equation for the restriction(s) of the problem. (0.5)
        - What is the branching factor (b) and the maximum depth (m)? Give an explanation of your answer. (0.5)
    - Comparison of the algorithms. (1 point)
        - Comparison of the time expended by the algorithms. (0.5)
        - Comparison of the space used in memory at a time by the algorithms. (0.5)

Part 2 - Shortest path from A to B:

1.
    - Explanation of the problem. (2 points)
        - Give the representation of a solution (answer) of the problem, as explained during the course. (0.5)
        - Give the equation of **f(n)** used in **Greedy Best-first Search** (or explain how to calculate f(n)). (0.5)
        - Give the equation of **f(n)** used in **A*** (or explain how to calculate f(n)). (0.5)
        - Explain both algorithms and the differences between them (0.5)

Conditions to approve assignment 1

- You have to have presented and defended your codes to a teacher.

- You have to send in your codes as txt files in Assignment 1 part 1 - Code and Assignment 1 part 2 - Code sections

    - Just take all your relevant code and copy paste into a txt file that you send in.
- Minimum score to pass assignment 1: 3 points.
    - Divided as a minimum of 2 points for part 1, and 1 point for part 2
- 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

ADDITIONAL INFORMATION!

You are allowed to send each assignment a maximum of three times. When you're approved in the assignment, you are not allowed to submit the assignment again.

Before submitting the report, you should present the assignments to a lab assistant. After that, and only if everything is correct, your report will be considered for grading i.e. you cannot get the points of the report without presenting the code.