Exercise 1. A* and hill climbing to find a path (Homework: a) and b) only)
The graph below corresponds to travelling between cities where the nodes are the cities and the links are the paths between them. S is the start city and G is the goal city. The step costs of travelling between the cities are shown on the links.

In addition, there is a heuristic value \( h(n) \) associated with each node \( n \):

<table>
<thead>
<tr>
<th></th>
<th>S</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>6</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

a) Run the hill-climbing algorithm to reach the goal G starting from S. Use the function \( v = \) smallest step cost. What is the path? What is its cost?

Note: In this case the evaluation function \( v \) is not associated with nodes but arcs, so you can’t check if the child node is better than the parent node, so ignore this check, i.e. assume that the child is always better than the parent.

b) Run the hill-climbing algorithm to reach the goal G starting from S. Use the function \( v(n) = \) smallest \( h(n) \).

Note: You need to check if the child node is better than the parent node.

c) Run A* to reach G from S.

1. Check if the heuristic is consistent and show your calculations.
2. Based on the result, run A* with re-opening or not reopening of closed nodes (your goal is to find the optimal solution, i.e. least cost path from S to G, and to expand as few nodes as possible). Among nodes with the same priority in the fringe, select the first added. Show the expended list with the f-values, the solution path and its cost.

Exercise 2. Hill-climbing and beam search (based on example from Winston, Artificial Intelligence, 1993, Addison-Wesley)
The undirected graph below depicts the step cost of traveling between cities. S is the start and G is the goal city.
The heuristic values $h$ from each node to the goal node G are shown below:

<table>
<thead>
<tr>
<th>S</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>10.4</td>
<td>6.7</td>
<td>4</td>
<td>8.9</td>
<td>6.9</td>
<td>3.0</td>
<td>0</td>
</tr>
</tbody>
</table>

a) Run the hill-climbing algorithm starting from S. Use the function 1) $v(n) =$ smallest step cost (the same note as in exercise 1a applies) and 2) $v = h$-value; the smaller the value, the better. List the expanded nodes. Was the goal node found?

b) Run beam search with $k=2$ using the $h$-values as an evaluation function (the smaller, the better). Do not re-visit the start node S and the already visited nodes. Was the goal node found?

**Exercise 3. Genetic algorithms** (adapted from Dunham, Data Mining, 2003, Pearson Education, ex.3.7)

Given an initial population $\{101010, 001100, 010101, 000010\}$, apply the genetic algorithm to find a better population.

Suppose the following:

**Fitness function:** It is defined as the sum of the bit values of each individual.

**Selection:** At each iteration select for crossover: 1) the best and the second best individual and 2) the best individual and the third best individual. If there are ties, resolve them randomly.

**Crossover:** Always choose the same crossover point, between the 3rd and 4th bit.

**Mutation:** Mutation means negating a bit. Always mutate bit 2.

**Stopping condition:** The average fitness for the entire population is greater than 4.

How many iterations were needed?

**Exercise 4. Local search (Advanced only)**

Give the name of the algorithm that results from each of the following special cases. If there is no exact equivalent, give the name of the algorithm that is most similar.

a) beam search with $k=1$

b) beam search with 1 initial state and no limit on the number of states retained.

c) simulated annealing with $T=\infty$ at all times

d) genetic algorithm with population size $N=1$

**Exercise 5. A* (Advanced only)**

Proof that if the $f$-values never decrease along any path to a goal, the heuristic is consistent. Hint: see the proof of the opposite (if the heuristic is consistent, than the $f$-value never decreases) from the lecture notes.

**Exercise 6. A**

Algorithm A * does not terminate until a goal node is selected for expansion from the fringe. However, a path to a goal node might be reached long before the node is selected for expansion, i.e. there may be already a goal node in the fringe. Why not terminate as soon as a goal node has been found (i.e. is in the fringe)? Illustrate your answer with an example.

**Exercise 7. A* - Admissibility and consistency of heuristics**

Last week we answered this question:

Does an admissible heuristic $h$ lead to non-decreasing $f$ values along any path? In other words, are all admissible heuristics consistent?
The answer is no, see the example below: h is admissible as h(n1)=4, which is not higher than the true cost to G (4); h(n2)=2 which is not higher than the true cost to G (3). However, h is not consistent as f(n1)=7 < f(n2)=6.

\[ \Rightarrow \text{admissibility (h)} \neq \Rightarrow \text{decreasing f-values (i.e. consistent heuristic)} \]

Recall also that: decreasing f-values \( \Leftrightarrow \) triangle equation for h

Question: Does an admissible heuristic \( h \) when combined with pathmax equation lead to non-decreasing \( f \) values along any path? For pathmax, see the lecture slides from this week.

**Exercise 8. A* search (to be done at your own time at home)**

Given is the following graph where the step costs are show along the links and the heuristic values of the nodes are in brackets. The start node is \( s \) and the goal nodes are \( h \) and \( i \).

b) Run A* and show the list of expanded nodes and the solution path. If there are nodes for expansion with the same priority, expand the last added node first.

c) Is the heuristic admissible? If it is not, correct it and show the new list of expanded nodes and the solution path.