Grading form assignment 1 - Algorithms & Data Structures

Group number: 181

Programming

Try running the **public** unit tests and verify that all of them are passed through a **correct implementation** (that is, make sure that the answers are not hard-coded). If a public unit test does not pass, or the implementation is not what it should be (for example if they were supposed to program a recursive algorithm but did something else), the assignment will not be graded, and a retake can be performed to obtain a grade.

Correctness

- ☑ (4 points) The *public unit* tests are working through a good implementation (and not by cheating the unit tests by e.g., hard-coding answers).

Number of failed private unit tests:	Points for the private unit tests (1-num_failed*0.125)
0	1

Additional feedback (if any, which private unit tests failed?)

Efficiency and coding style

	Feedback	**Points given**	**Max number of points reachable**
Variable names (informative, lowercase: not CamelCase but instead camel_case)		**0.33**	0.33
Efficiency of solutions (not more expensive or complicated than required)	**There are too many unnecessary functions**	**0**	0.33
Useful comments in functions to explain what is happening	**You don't need to comment every lines, you can do it with naturally**	**0.33**	0.33

	occurring groups of code. But the comments do make the code very clear :)		

Report

Introduction

- ☑ (0.5 points) It is explicitly stated that the “state” is a (partial) filling of the 2D grid
- ☐ (0.5 points) It is explicitly stated that an “action” is filling in a value of a given cell in the 2D grid

Additional feedback

I see that the action is explained but you never state that this is an action :(

Difference exhaustive search & Backtracking

- ☑ (0.33 points) It is explained that exhaustive search simply enumerates all possible states and checks whether <u>completely filled states</u> satisfy the constraints of the problem and thus are solutions to the problem.
- ☑ (0.33 points) It is explained that backtracking builds a solution step-by-step and after every action checks whether it is still possible to arrive at a solution from the partial state. If not, the search is cut off and we continue along another branch.
- ☑ (0.33 points) It is explained that exhaustive search can be transformed into backtracking by adding a check after every action to check whether the partial state can still lead to a valid solution

Additional feedback

Search tree

- ☑ (0.25 points) A correct search tree has been drawn for exhaustive search
- ☑ (0.5 points) A correct search tree for backtracking has been drawn for the same problem that clearly illustrates the difference between the approaches
- ☑ (0.25 points) It is observed that backtracking is still optimal as it only discards candidates that can never lead to a valid solution

Additional feedback

Clean looking trees :)

Greedy approach

- ☑ (0.25 points) A greedy approach is proposed that makes locally optimal decisions
- ☐ (0.25 points) The explanation of whether their approach is optimal is logical and correct
- ☐ (0.5 points) The provided adversarial test case is correct and displays that the proposed greedy method is not optimal. The points can also be given if they have actually proposed an optimal greedy method (in which case an adversarial example is not needed).

Additional feedback

Very nice graph with run time and everything, but optimal refers to if the algorithm finds all the solutions or not, so the greedy approach would be optimal if it reaches every possible correct state.

Final grade: 8.4