SCALE FOR PROJECT N-PUZZLE (/PROJECTS/N-PUZZLE)

Introduction

In order to have a productive and tolerable grading session, we ask that you :

- Stay courteous, polite, respectful and constructive during this session.
The bond of trust between members of the 42 community depends on it;
- Take care to show the graded person(s) the problems you notice, and
explain them as best you can;
- Accept that there may be differences in interepretation on the featureset
and/or what the subject requires. Stay open-minded, try to honestly
determine who is right and who is not, and grade accordingly.

Guidelines

Remember that you must ONLY grade what's on the turn-in repository !

You have to "git clone" the repository, and grade what's on it,
AND ONLY WHAT IS ON IT.

Attachments

Subject (https://cdn.intra.42.fr/pdf/pdf/2026/npuzzle.en.pdf)

Puzzle generator (/uploads/document/document/99/res_npuzzle-gen.py)

First and foremost

Preliminary checks

Check the following elements:

- There is something in the git repository

- The "auteur" file, if required by the subject, is present and valid
- The Makefile, if required, is present and has the required rules

If one of these elements is not in confirmity with what the subject requires, the session stops. You may still debate on the project, but you are not to grade the student(s).

During the rest of this session, if the program has an inappropriate behaviour (Segfault, bus error, double-free, uncaught exception, etc ...), the session stops.

☑ Yes ☒ No

Actually running the program

Output correctness

The students have the output required by the subject, which is:
- Complexity in time
- Complexity in size
- Number of moves from initial state to solution
- Ordered sequence of states that make up the solution

Or, if the puzzle is unsolvable, it has to be displayed, and the program has to exit cleanly.

☑ Yes ☒ No

Solution

Test the program on some puzzles, some from the students (the subject requires that they bring some) and some using the generator in the subject.

The program has to find a solution when it is possible, and if the puzzle is unsolvable, it has to tell the user and exit cleanly.
The solution has to be valid : Check that the program does not "cheat".

If the generator says the puzzle is solvable, then it IS solvable, so no excuses.

>>> IF YOU DO NOT ANSWER YES TO THIS QUESTION, THE SESSION ENDS, DO NOT GRADE THE REST. <<<

☑ Yes ☒ No

Bare-minimum time efficiency

The program is able to find a solution to a solvable 3-puzzle in under a few seconds. More than 10 seconds is unacceptable.

☑ Yes ☒ No

Search algorithm

A* algorithm

The students implemented the A* algorithm or one of its variants (B*, IDA*, etc ...) AND are able to explain it comprehensibly.

☑ Yes ☒ No

Heuristics

The students let the user choose between at least 3 (relevant) heuristic functions AND are able to explain how their heuristics work AND are able to explain why their heuristics are admissible.

☑ Yes ☒ No

Data structures1

Open set

The students have implemented their "open" set with
a relevant data structure.
- A priority queue (i.e a queue that gives you the high-priority elements first, here "high-priority" means "low-cost"), or another container that allows for immediate retrieval of the lowest-cost item, is worth 5
- A map/dict/hash of (cost) -> (node list) is worth 3
- Another container with a kind of mechanism to easily find the highest priority element is worth 2

- Any other container, such as list/vector/etc ... is worth 0

Also, the students must be able to explain the relevance of their choice.
If they can not, this is worth 0

Rate it from 0 (failed) through 5 (excellent)

Closed set

The students have implemented their "closed" set with
a relevant data structure.
- A container that easily allows to check whether a node currently
is in the set or not is worth 5
- Any other container, such as list/vector/etc ... is worth 0

Also, the students must be able to explain the relevance of their
choice. If they can not, this is worth 0

Rate it from 0 (failed) through 5 (excellent)

Data structures2

Greedy search1

The students have implemented an option to do a greedy search and are able to explain it.

☑ Yes ☒ No

Greedy search2

The students have implemented an option to do a uniform-cost search and are able to explain it.

☑ Yes ☒ No

Other interesting bonuses

The students have implemented other interesting bonuses. 1 point per identifiable bonus feature that you deem relevant and/or interesting.

Rate it from 0 (failed) through 5 (excellent)

Ratings

Don't forget to check the flag corresponding to the defense

⊘ Forbidden function

Conclusion

Leave a comment on this evaluation

Preview!!!