Introduction to Artificial Intelligence
Assignment 2: Sudoku Solver

Deadline

Week 14 (01.12.24 23:59)

Submission

1. Only one source code file has to be uploaded to Codeforces. Codeforces handle doesn't have to be uploaded to Moodle, if was sent during ITP/SSAD/TCS courses and not changed after (otherwise, contact course instructor)
2. Report has to be uploaded to Moodle with the file name like *NameSurname.pdf,* e.g., *IvanIvanov.pdf*. No other symbols allowed

Programming Languages

Java (JDK 8 or 21) or C++ (C++17 or 20 or 23) or C# (8 or 10) or Kotlin (1.7 or 1.9) or Python 3 (not recommended)

Requests

- The program must work, the code should be readable, well-structured and should contain English comments in a language-specified format
- It has to be only one code file
- It has to be only one report **.pdf* file
- It is allowed to use only standard libraries
- NO extension of a deadline. Works sent after the deadline will NOT be evaluated
- Excused students can have extension of deadline by number of excused days intersecting with assignment days based on DoE's information. Still, the deadline can't be pushed later than 07/12/24
- Assignment is individual
- We will be using MOSS (Measure of Software Similarity) as a test for plagiarism. Be reminded that a score of 0 will be assigned to any submissions suspected of plagiarism pending a full investigation as per IU policies

Grading Criteria

- 60% for the code correctness
- 10% for code readability and comments quality
- 30% for the well-structured and informative report

Task

Your search space is a 9*9 Sudoku grid with some numbers initially filled in, and your task is to solve the Sudoku. You are given a table containing 81 symbols: single-digit numbers (givens) and hyphens separated by spaces, which have to be used for Sudoku solving. Known numbers (givens) can't be modified, hyphens have to be replaced by single-digit numbers. By using an evolutionary algorithm (EA), you have to solve Sudoku following several rules:

1. Each of 9 rows should contain 9 different single-digit numbers
2. Each of 9 columns should contain 9 different single-digit numbers
3. Each of 9 3*3 subgrid should contain 9 different single-digit numbers

Correct Sudoku should be having a single solution. In this assignment only correct ones will be provided. You are allowed to use any type of EA, however, you are obliged to use both crossover and mutation for evolving Sudoku.

Report

Your report should deeply describe chosen EA algorithm, its flow, fitness function, specifics of variation operators and EA parameters in plain English. Statistics demonstrating the average and maximum population fitness at final generations for different number of givens or complexity levels should be provided. It may be said that the complexity of Sudoku depends on the number of givens and techniques required to solve Sudoku. There are different classifications of complexity. One of them divides levels of complexity into *easy*, *medium*, *hard* and *expert*. You may consider any of the existing classifications to run your own tests. Do not forget to attach a reference to the classification idea.

The results of statistics have to be supported by generated plots. It means that for X-axis you should have number of givens or complexity level (based on your choice) and for Y-axis average for max/avg fitness on last generations among all tests. You are supposed to use your own tests to provide statistics.

If you prefer to use number of givens for X-axis, then assume that for each number of givens there should be at least 10 tests. The biggest number of given numbers, which should be used is 40, the smallest is 20. The more numbers are known, the easier the task in general.

If you prefer to use levels of complexity for X-axis, then for each level there should be at least 50 tests with different number of givens preferably.

Evaluation

Convergence of EAs for the same task may be varying in terms of time and generations. You should consider Codeforces limitation per test, which is 30 seconds. However, we understand that it is not a trivial task to implement fast algorithm. Moreover, time used by algorithm depends on the language speed, that is why Python may be not the best solution for this task.

For the students, whose codes won't fit in Codeforces limitation, it is planned to provide another way of grading, which will be announced later based on average progress reported by students in the chat. However, waiting for too long should not be expected for another way of grading. Assume that codes evaluated via Codeforces will bring extra bonus point for the course.

Inputs

The input represented by *input.txt* file ending with a new line character. The input is guaranteed to be giving 1 valid solution. The input contains 9 rows of single-digit numbers (1-9) and hyphens separated by spaces. Each row contains 9 elements. First row represents the first row in Sudoku, second 9 row represent the second row, etc. Example of the input is provided on Figure 1.

```
- 8 -   - - -   - 9 -
- - 7   5 - 2   8 - -
6 - -   8 - 7   - - 5

3 7 -   - 8 -   - 5 1
2 - -   - - -   - - 8
9 5 -   - 4 -   - 3 2

8 - -   1 - 4   - - 9
- - 1   9 - 3   6 - -
- 4 -   - - -   - 2 -
```

Figure 1. Input example

Outputs

The output is represented by *output.txt* file ending with a new line character. It should contain 81 single-digit numbers (1-9), including unmodified given numbers separated by spaces. Each next 9 numbers should be put on a new line.

Example

In Figure 2 the graphical example of one of the used tests is represented.

	8						9	
		7	5		2	8		
6			8		7			5
3	7			8			5	1
2								8
9	5			4			3	2
8			1		4			9
		1	9		3	6		
	4						2	

1	8	5	4	3	6	2	9	7
4	3	7	5	9	2	8	1	6
6	9	2	8	1	7	3	4	5
3	7	6	2	8	9	4	5	1
2	1	4	3	7	5	9	6	8
9	5	8	6	4	1	7	3	2
8	6	3	1	2	4	5	7	9
7	2	1	9	5	3	6	8	4
5	4	9	7	6	8	1	2	3

Figure 2. Example