Вештачка интелигенција-2021/2022/L

Quiz navigation

1 2

Finish attempt ...

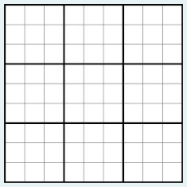
Question **2**

Correct

Mark 3.00 out of 3.00

Flag question

Дадена ви е играта судоку. Во оваа игра во секој блок се стават бројки од 1 до 9 така што во ниедна редица, колона и блок не смее да се повторува ниедна цифра. Почетно сите полиња се празни. Вашата задача е да најдете решение на овој проблем. Просторот ви е даден на сликата подолу.



Забелешка: На влез добивате со каков `Solver` да работите. Испечатете го само првото решение. **Не е задолжително да ви поминуваат сите тест примери за задачата да биде точна. Зависи како сте ги поставиле условите.**

Потсетник: Во даденот модул constraint веќе се имплементирани следните ограничувања како класи: *AllDifferentConstraint, AllEqualConstraint, MaxSumConstraint, ExactSumConstraint, MinSumConstraint, InSetConstraint, NotInSetConstraint, SomeInSetConstraint, SomeNotInSetConstraint.*

For example:

Input	Result
BacktrackingSolver	{0: 9, 1: 8, 2: 7, 3: 6, 4: 5, 5: 4, 6: 3, 7: 2, 8: 1, 9: 6, 10: 5, 11: 4, 15: 9, 16: 8, 17: 7, 12: 3, 13: 2, 14: 1, 18: 3, 19: 2, 20: 1, 21: 9, 22: 8, 23: 7, 24

Answer: (penalty regime: 0 %)

Reset answer

```
from constraint import *

if __name__ == '__main__':
    problem = Problem(BacktrackingSolver())
    solver = input()

    if solver == "BacktrackingSolver":
        problem = Problem(BacktrackingSolver())
    elif solver == "MinConflictsSolver":
        problem = Problem(MinConflictsSolver())
    elif solver == "RecursiveBacktrackingSolver":
        problem = Problem(RecursiveBacktrackingSolver())

    domain = range(1, 10)  # numbers from 1 to 9

    variables = [i for i in range(0, 81)]  # The sudoku game area is 9x9 or 81 fields
    for variable in variables:
        problem.addVariable(variable, domain)

    # ---Tuka dodadete gi ogranichuvanjata----------

    # rows and columns constraint:
    for i in range(9):
        problem.addConstraint(AllDifferentConstraint(), [i * 9 + num for num in range(9)])
        problem.addConstraint(AllDifferentConstraint(), [i + 9 * num for num in range(9)])

    # constraint for every 3 x 3 square:
    for row in range(0, 9, 3):
        for col in range(0, 9, 3):
            problem.addConstraint(AllDifferentConstraint(),
                                  [i * 9 + i for i in range(row, row + 3) for i in range(col, col + 3)])
```

Check

	Input	Expected
✔	BacktrackingSolver	{0: 9, 1: 8, 2: 7, 3: 6, 4: 5, 5: 4, 6: 3, 7: 2, 8: 1, 9: 6, 10: 5, 11: 4, 15: 9, 16: 8, 17: 7, 12: 3, 13: 2, 14: 1, 18: 3, 19: 2, 20: 1, 21: 9, 22: 8, 23:

Passed all tests! ✔

Correct

Marks for this submission: 3.00/3.00.

Previous page

Finish attempt ...

⇥ Тест 2

Jump to...

Класична лабораториска вежба 1 ⇤