Zadanie skúšky 18.1.2016

Sudoku

Zadefinujte triedu **Sudoku**, ktorá nám pomôže riešiť jednoduché zadania hlavolamu sudoku. Samotné sudoku je štvorcová sieť 9x9 políčok, pričom, buď je políčko prázdne (budeme ho reprezentovať nulovou hodnotou), alebo obsahuje číslo od 1 do 9. Pre takúto hraciu plochu platí:

- v každom riadku sa žiadne číslo 1 až 9 nevyskytuje viackrát;
- v každom stĺpci sa žiadne číslo 1 až 9 nevyskytuje viackrát;
- celú štvorcové sieť ešte rozdelíme na 9 menších štvorcov veľkosti 3x3 políčok: aj pre každý takýto štvorec sa žiadne číslo 1 až 9 nevyskytuje viackrát.

Na začiatku zadania hlavolamu sudoku môžu byť niektoré políčka prázdne a úlohou hráča je potom doplniť na všetky tieto políčka čísla 1 až 9 tak, aby ostali splnené pravidlá sudoku. Trieda **Sudoku** nedokáže vyriešiť každú zadanie, ale len niektoré veľmi jednoduché. Bude pracovať takouto metódou:

- pre každé prázdne políčko vie zistiť množinu všetkých čísel, ktoré by sme sem mohli zapísať bez porušenia pravidiel – zrejme táto množina je podmožinou množiny čísel od 1 do 9
- zrejme, ak je táto množina pre nejaké prázdne políčko prázdna, sudoku sa vyriešiť nedá;
- ak je ale táto množina jednoprvková, môžeme tento jeden prvok na toto políčko zapísať a tým sa priblížiť k vyriešeniu hlavolamu
- ak budeme postupne zo všetkých jednoprvkových množín zapisovať prvky do príslušných políčok štvorcovej siete, môže sa nám takto podariť vyriešiť aj celý hlavolam.

Počiatočné rozloženie čísel je zadané vo vstupnom súbore:

- každý riadok súboru popisuje jedno neprázdne políčko štvorcovej siete;
- sú to tri čísla: poradové číslo riadku, poradové číslo stĺpca a hodnota na tomto políčku, poradové čísla sú od 0 do 8 a hodnota políčka je číslo od 1 do 9

Zadefinujte triedu **Sudoku**:

```
class Sudoku:
    def __init__(self, meno_suboru):
        self.pole = []
        ...

    def __repr__(self):
        ...

    def zisti(self):
        ...

    def __getitem__(self, pozicia):
        ...

    def __setitem__(self, pozicia, hodnota):
        ...

    def krok(self):
        ...

    def ries(self):
        ...
```

kde

- metóda **`__init__(self, meno_suboru)`:** prečíta súbor a vytvorí dvojrozmerné pole (atribút **`pole`**) s počiatočným rozložením čísel (prázdne políčka budú mať hodnotu 0);
- metóda **`__repr__(self)`:** vráti znakový reťazec, ktorý reprezentuje momentálny stav štvorcovej siete: medzi políčkami v riadku je znak medzera, prázdne políčka sú znak **`'.'`** a medzi riadkami je znak **`'\n'`**;
- metóda **`zisti(self)`:** vráti jednu z týchto hodnôt: **`0`** označuje chybne zadanú plochu (nejaké číslo sa v riadku, stĺpci alebo štvorci 3x3 opakuje viackrát), **`1`** označuje korektné zadanie, ktoré obsahuje prázdne políčka, **`2`** označuje kompletne vyriešený hlavolam bez prázdnych políčok;

- metóda **`__getitem__(self, pozicia)`**: vráti pre prázdne políčko na zadanej pozícii (parameter **`pozicia`** je dvojica **`(riadok, stĺpec)`**) **množinu** prípustných hodnôt, pre neprázdne políčko vráti **`None`**;
- metóda **`__setitem__(self, pozicia, hodnota)`**: priradí na zadanú pozíciu danú hodnotu (parameter **`pozicia`** je dvojica **`(riadok, stĺpec)`**) – metóda modifikuje atribút **`pole`**;
- metóda **`krok(self)`**: postupne prejde všetky políčka siete (po riadkoch) a skontroluje príslušné množiny prípustných hodnôt: ak je to jednoprvková množina, políčko sa nahradí touto hodnotou, ak je to prázdna množina, metóda skončí a vráti hodnotu -1, ak takto prejde všetkých 81 políčok, metóda vráti celkový počet nahradených políčok (t.j. počet políčok, ktoré mali jednoprvkové množiny);
- metóda **`ries(self)`**: veľa krát volá metódu **`krok()`**, kým sa ešte nejaké prázdne políčka zapĺňali novými hodnotami, metóda vráti **`False`**, keď sa počas zapĺňania objavila nejaká prázdna množina (metóda **`krok()`** vrátila **`-1`**), inak metóda vráti **`True`**

Napr. pre takéto zadanie sudoku:

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

tento test:

```
if __name__ == '__main__':
    s = Sudoku('subor.txt')
    print(s)
    print('mnozina pre 0,0 je', s[0,0])
    print('mnozina pre 7,0 je', s[7,0])
    if s.zisti() == 0:
        print('chybne zadanie')
    else:
        if s.ries():
            print(s)
        else:
            print('nema riesenie')
```

vypíše:

```
. . . . . 6 8 . 7
8 . 7 5 . 3 2 4 .
2 6 5 8 7 . . 3 .
5 8 . . . . 7 2 .
. 7 . 2 4 8 . 6 5
4 . 2 7 . 5 . 8 .
7 . 4 . . 2 . 9 8
. 5 1 3 8 9 . 7 .
9 . . . . 7 6 . 3
mnozina pre 0,0 je {1, 3}
mnozina pre 7,0 je {6}
. . . . . 6 8 . 7
8 . 7 5 . 3 2 4 .
2 6 5 8 7 4 . 3 .
5 8 . . . 1 7 2 .
. 7 . 2 4 8 . 6 5
4 . 2 7 . 5 . 8 .
7 3 4 . . 2 . 9 8
6 5 1 3 8 9 4 7 2
9 2 8 . . 7 6 . 3
```

Uvedomte si, že volanie **`s[0,0]`** spôsobí zavolanie metódy **`s.__getitem__((0, 0))`**.

Aby ste mohli spúšťať skúškové testy, program uložte do súboru **skuska.py**. Riešenie (bez dátových súborov) odovzdajte na úlohový server **http://capek.ii.fmph.uniba.sk/list**.

Skúška pokračuje od 12:00 vyhodnotením v kancelárii **m162**.