

به نام خدا

لعیا فاخر ۹۸۲۳۹۴۳

## حل مساله مربع لاتین با استفاده از روش ارضای محدودیت

الگوریتم کلی این مساله به این صورت است که ابتدا عدد  $N$  دریافت شده و یک ماتریس مربعی  $N$  در  $N$  ساخته میشود. سپس یک گراف محدودیت ایجاد می شود به طوری که هر خانه در ماتریس متناظر با یک نود در گراف و بین هر خانه ی ماتریس با خانه هایی که با آن ها در یک سطر و یا در یک ستون است یک یال قرار داده میشود. (در اصل مساله ی ما به یک مساله ی رنگ آمیزی نقشه تبدیل می شود). سپس توسط الگوریتم های ارضای محدودیت از جمله `min_conflict` و `Backtracking` و `Forward checking` مساله حل شده و مربع لاتین خروجی به همراه زمان اجرا و تعداد بررسی های انجام شده در هر الگوریتم، در کنسول نمایش داده می شود.

فایل `main.py`:

در تابع ابتدا مقدار  $N$  از کاربر دریافت شده و توسط تابع `build_matrix` مربع لاتین در ابعاد  $N$  در  $N$  با شماره ی ایندکس های ۱ تا  $N*N$  ایجاد می شود و مقدار آن برگردانده می شود. سپس توسط تابع `generate_graph` و `inARowOrCol` یک لیست برای ایجاد روابط در گراف محدودیت ساخته می شود. در این تابع لیست محدودیت ها (لیست همسایه ها یا `borders`) به صورت تعدادی زوج مرتب ذخیره و برگردانده میشوند به طوری که اگر دو خانه در ماتریس، در یک سطر یا ستون باشند به صورت یک زوج مرتب در این لیست ذخیره می شوند. بعداً در کلاسها و متدهای دیگر که به آنها می پردازیم، این لیست به یک گراف تبدیل شده و مقدار دهی خانه های مربع انجام می شود. (طبق مساله ی مربع لاتین می دانیم که اعداد هر سطر و اعداد هر ستون باید متفاوت باشند بنابراین این در گراف محدودیت باید بین آن خانه ها یک یال قرار داده شود چون مانند همسایه ها در مساله ی رنگ آمیزی نقشه، نباید مقدار برابر داشته باشند. در این مساله خانه های مربع مانند مناطق و اعداد ۱ تا  $N$  مانند رنگ ها در مساله ی رنگ آمیزی نقشه فرض شده اند. خانه های مربع لاتین باید به گونه ای مقدار دهی شوند که مقدار آنها با همسایه ها (خانه هایی که در سطر یا ستون یکسان هستند) متفاوت باشد. بنابراین این خانه هایی که در یک سطر یا ستون مشترکند نباید شماره ی برابر داشته باشند).

در نهایت در تابع `main` توابع `MinConflictSolution` و `backtracking_solution` از فایل `csp_solution.py` فراخوانده می شوند و مربع لاتین خروجی به همراه زمان اجرای هر الگوریتم و تعداد بررسی های انجام شده در هر الگوریتم، در خروجی نمایش داده می شود.

فایل `csp_solution.py`:

در این فایل کلاس هایی به نام `solution` و `backtracking_solution` و `MinConflictSolution` قرار دارند که الگوریتم های متناظر آنها پیاده سازی شده است.

در کلاس RecursiveBacktrackingSolver اگر مقدار بولین forwardcheck برابر با True باشد، از forwardchecking برای بهینه سازی در حل مساله استفاده می شود، در غیر این صورت فقط از Backtracking استفاده میشود. در این تابع از هیوریستیک کمترین مقدار باقی مانده (Minimum Remaining Values) استفاده شده است.

فایل graph.py :

در این فایل یک کلاس با همین نام قرار دارد و برای مدلسازی مساله و تعیین متغیر های مقدار دهی شده و مقدار دهی نشده و ایجاد محدودیت های بین متغیر ها و همچنین راه حل استفاده شده برای حل و به عبارتی برای مدلسازی گراف محدودیت، پیاده سازی شده است. در تابع main به ازای هر یک از الگوریتم های حل مساله، این گراف محدودیت ایجاد می شود. یکی از مهم ترین متد ها در این کلاس printsolutions است که مربع لاتین حل شده را در خروجی چاپ میکند.

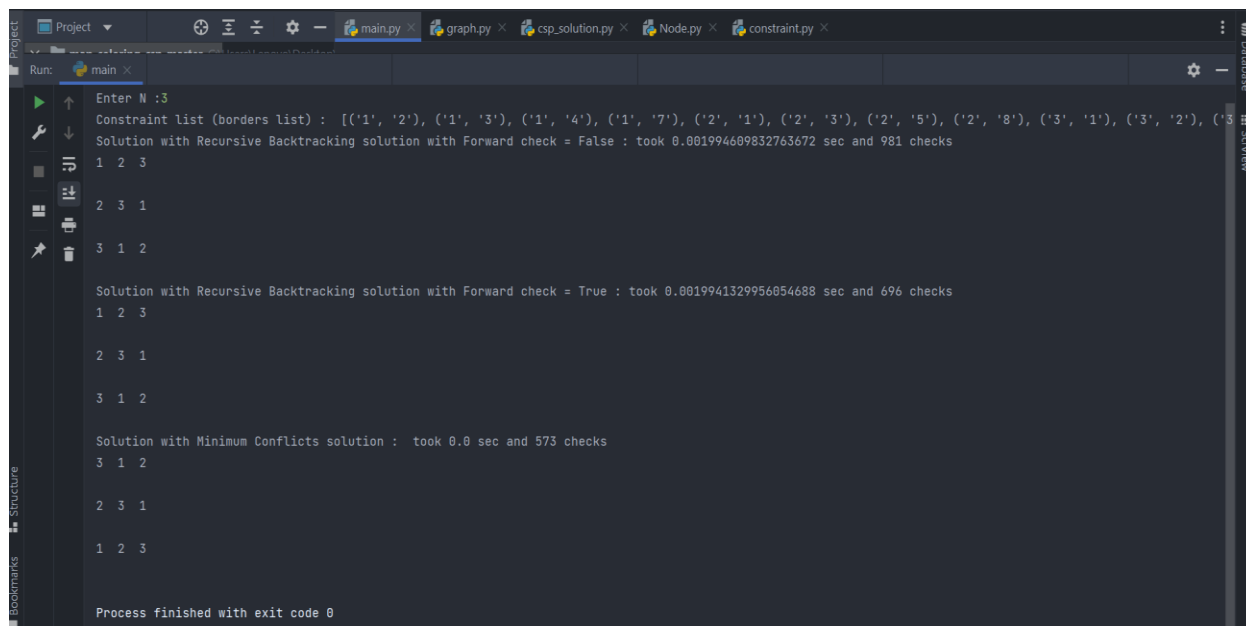
فایل Node.py:

در این فایل دو کلاس به نام های Node و Domain قرار دارند. نود های گراف و همچنین دامنه ی مجاز آنها توسط این دو کلاس با استفاده از لیست محدودیت مدلسازی می شوند. از این کلاس در فایل و کلاس graph استفاده شده است.

فایل constraint.py :

در این کلاس محدودیت ها برای مقادیر مجاز متغیر هایی که تا کنون مقدار دهی نشده اند بررسی می شود. بدنه ی متد forward\_check نیز در این کلاس پیاده سازی شده است.

نمونه هایی از ورودی و خروجی های برنامه به ازای مقادیر مختلف N:



```
Enter N :3
Constraint list (borders list) : [('1', '2'), ('1', '3'), ('1', '4'), ('1', '7'), ('2', '1'), ('2', '3'), ('2', '5'), ('2', '8'), ('3', '1'), ('3', '2'), ('3', '4'), ('3', '5'), ('3', '6'), ('3', '9')]
Solution with Recursive Backtracking solution with Forward check = False : took 0.001994609832763672 sec and 981 checks
1 2 3
2 3 1
3 1 2
Solution with Recursive Backtracking solution with Forward check = True : took 0.0019941329956054688 sec and 696 checks
1 2 3
2 3 1
3 1 2
Solution with Minimum Conflicts solution : took 0.0 sec and 573 checks
3 1 2
2 3 1
1 2 3
Process finished with exit code 0
```

```
Enter N :2
Constraint list (borders list) : [('1', '2'), ('1', '3'), ('2', '1'), ('2', '4'), ('3', '1'), ('3', '4'), ('4', '2'), ('4', '3')]
Solution with Recursive Backtracking solution with Forward check = False : took 0.0 sec and 38 checks
1 2

2 1

Solution with Recursive Backtracking solution with Forward check = True : took 0.00101470947265625 sec and 32 checks
1 2

2 1

Solution with Minimum Conflicts solution : took 0.0 sec and 51 checks
1 2

2 1

Process finished with exit code 0
```

```
Enter N :4
Constraint list (borders list) : [('1', '2'), ('1', '3'), ('1', '4'), ('1', '5'), ('1', '9'), ('1', '13'), ('2', '1'), ('2', '3'), ('2', '4'), ('2', '6'), ('2', '10'), ('3', '1'), ('3', '2'), ('3', '4'), ('3', '5'), ('3', '6'), ('3', '7'), ('3', '8'), ('3', '9'), ('3', '10'), ('3', '11'), ('3', '12'), ('3', '13'), ('4', '1'), ('4', '2'), ('4', '3'), ('4', '5'), ('4', '6'), ('4', '7'), ('4', '8'), ('4', '9'), ('4', '10'), ('4', '11'), ('4', '12'), ('4', '13'), ('5', '1'), ('5', '2'), ('5', '3'), ('5', '4'), ('5', '5'), ('5', '6'), ('5', '7'), ('5', '8'), ('5', '9'), ('5', '10'), ('5', '11'), ('5', '12'), ('5', '13'), ('6', '1'), ('6', '2'), ('6', '3'), ('6', '4'), ('6', '5'), ('6', '6'), ('6', '7'), ('6', '8'), ('6', '9'), ('6', '10'), ('6', '11'), ('6', '12'), ('6', '13'), ('7', '1'), ('7', '2'), ('7', '3'), ('7', '4'), ('7', '5'), ('7', '6'), ('7', '7'), ('7', '8'), ('7', '9'), ('7', '10'), ('7', '11'), ('7', '12'), ('7', '13'), ('8', '1'), ('8', '2'), ('8', '3'), ('8', '4'), ('8', '5'), ('8', '6'), ('8', '7'), ('8', '8'), ('8', '9'), ('8', '10'), ('8', '11'), ('8', '12'), ('8', '13'), ('9', '1'), ('9', '2'), ('9', '3'), ('9', '4'), ('9', '5'), ('9', '6'), ('9', '7'), ('9', '8'), ('9', '9'), ('9', '10'), ('9', '11'), ('9', '12'), ('9', '13'), ('10', '1'), ('10', '2'), ('10', '3'), ('10', '4'), ('10', '5'), ('10', '6'), ('10', '7'), ('10', '8'), ('10', '9'), ('10', '10'), ('10', '11'), ('10', '12'), ('10', '13'), ('11', '1'), ('11', '2'), ('11', '3'), ('11', '4'), ('11', '5'), ('11', '6'), ('11', '7'), ('11', '8'), ('11', '9'), ('11', '10'), ('11', '11'), ('11', '12'), ('11', '13'), ('12', '1'), ('12', '2'), ('12', '3'), ('12', '4'), ('12', '5'), ('12', '6'), ('12', '7'), ('12', '8'), ('12', '9'), ('12', '10'), ('12', '11'), ('12', '12'), ('12', '13'), ('13', '1'), ('13', '2'), ('13', '3'), ('13', '4'), ('13', '5'), ('13', '6'), ('13', '7'), ('13', '8'), ('13', '9'), ('13', '10'), ('13', '11'), ('13', '12'), ('13', '13')]
Solution with Recursive Backtracking solution with Forward check = False : took 0.2702760696411133 sec and 264160 checks
1 4 3 2
3 2 4 1
4 1 2 3
2 3 1 4

Solution with Recursive Backtracking solution with Forward check = True : took 0.12566375732421875 sec and 56928 checks
1 4 3 2
3 2 4 1
4 1 2 3
2 3 1 4

Solution with Minimum Conflicts solution : took 0.0019948482513427734 sec and 1897 checks
3 2 4 1
1 4 3 2

Process finished with exit code 0
```

```
Solution with Minimum Conflicts solution : took 0.0019948482513427734 sec and 1897 checks
3 2 4 1

1 4 3 2

4 1 2 3

2 3 1 4

Process finished with exit code 0
```

```
Enter N :3
Constraint list (borders list) : [('1', '2'), ('1', '3'), ('1', '4'), ('1', '7'), ('2', '1'), ('2', '3'), ('2', '5'), ('2', '8'), ('3', '1'), ('3', '2'), ('3', '4'), ('3', '5'), ('3', '6'), ('3', '7'), ('3', '8'), ('3', '9'), ('4', '1'), ('4', '2'), ('4', '3'), ('4', '4'), ('4', '5'), ('4', '6'), ('4', '7'), ('4', '8'), ('4', '9'), ('5', '2'), ('5', '3'), ('5', '4'), ('5', '5'), ('5', '6'), ('5', '7'), ('5', '8'), ('5', '9'), ('6', '3'), ('6', '4'), ('6', '5'), ('6', '6'), ('6', '7'), ('6', '8'), ('6', '9'), ('7', '1'), ('7', '4'), ('7', '5'), ('7', '6'), ('7', '7'), ('7', '8'), ('7', '9'), ('8', '2'), ('8', '3'), ('8', '4'), ('8', '5'), ('8', '6'), ('8', '7'), ('8', '8'), ('8', '9'), ('9', '1'), ('9', '2'), ('9', '3'), ('9', '4'), ('9', '5'), ('9', '6'), ('9', '7'), ('9', '8'), ('9', '9')]
Solution with Recursive Backtracking solution with Forward check = False : took 0.0009856224060058594 sec and 981 checks
1 2 3
2 3 1
3 1 2

Solution with Recursive Backtracking solution with Forward check = True : took 0.0019941329956054688 sec and 696 checks
1 2 3
2 3 1
3 1 2

Solution with Minimum Conflicts solution : took 0.0 sec and 199 checks
2 3 1
1 2 3
3 1 2

Process finished with exit code 0
```

```
Enter N :4
Constraint list (borders list) : [('1', '2'), ('1', '3'), ('1', '4'), ('1', '5'), ('1', '6'), ('1', '7'), ('1', '8'), ('1', '9'), ('1', '10'), ('1', '11'), ('1', '12'), ('1', '13'), ('1', '14'), ('2', '1'), ('2', '3'), ('2', '4'), ('2', '5'), ('2', '6'), ('2', '7'), ('2', '8'), ('2', '9'), ('2', '10'), ('2', '11'), ('2', '12'), ('2', '13'), ('2', '14'), ('3', '1'), ('3', '2'), ('3', '4'), ('3', '5'), ('3', '6'), ('3', '7'), ('3', '8'), ('3', '9'), ('3', '10'), ('3', '11'), ('3', '12'), ('3', '13'), ('3', '14'), ('4', '1'), ('4', '2'), ('4', '3'), ('4', '4'), ('4', '5'), ('4', '6'), ('4', '7'), ('4', '8'), ('4', '9'), ('4', '10'), ('4', '11'), ('4', '12'), ('4', '13'), ('4', '14'), ('5', '1'), ('5', '2'), ('5', '3'), ('5', '4'), ('5', '5'), ('5', '6'), ('5', '7'), ('5', '8'), ('5', '9'), ('5', '10'), ('5', '11'), ('5', '12'), ('5', '13'), ('5', '14'), ('6', '1'), ('6', '2'), ('6', '3'), ('6', '4'), ('6', '5'), ('6', '6'), ('6', '7'), ('6', '8'), ('6', '9'), ('6', '10'), ('6', '11'), ('6', '12'), ('6', '13'), ('6', '14'), ('7', '1'), ('7', '2'), ('7', '3'), ('7', '4'), ('7', '5'), ('7', '6'), ('7', '7'), ('7', '8'), ('7', '9'), ('7', '10'), ('7', '11'), ('7', '12'), ('7', '13'), ('7', '14'), ('8', '1'), ('8', '2'), ('8', '3'), ('8', '4'), ('8', '5'), ('8', '6'), ('8', '7'), ('8', '8'), ('8', '9'), ('8', '10'), ('8', '11'), ('8', '12'), ('8', '13'), ('8', '14'), ('9', '1'), ('9', '2'), ('9', '3'), ('9', '4'), ('9', '5'), ('9', '6'), ('9', '7'), ('9', '8'), ('9', '9'), ('9', '10'), ('9', '11'), ('9', '12'), ('9', '13'), ('9', '14'), ('10', '1'), ('10', '2'), ('10', '3'), ('10', '4'), ('10', '5'), ('10', '6'), ('10', '7'), ('10', '8'), ('10', '9'), ('10', '10'), ('10', '11'), ('10', '12'), ('10', '13'), ('10', '14'), ('11', '1'), ('11', '2'), ('11', '3'), ('11', '4'), ('11', '5'), ('11', '6'), ('11', '7'), ('11', '8'), ('11', '9'), ('11', '10'), ('11', '11'), ('11', '12'), ('11', '13'), ('11', '14'), ('12', '1'), ('12', '2'), ('12', '3'), ('12', '4'), ('12', '5'), ('12', '6'), ('12', '7'), ('12', '8'), ('12', '9'), ('12', '10'), ('12', '11'), ('12', '12'), ('12', '13'), ('12', '14'), ('13', '1'), ('13', '2'), ('13', '3'), ('13', '4'), ('13', '5'), ('13', '6'), ('13', '7'), ('13', '8'), ('13', '9'), ('13', '10'), ('13', '11'), ('13', '12'), ('13', '13'), ('13', '14'), ('14', '1'), ('14', '2'), ('14', '3'), ('14', '4'), ('14', '5'), ('14', '6'), ('14', '7'), ('14', '8'), ('14', '9'), ('14', '10'), ('14', '11'), ('14', '12'), ('14', '13'), ('14', '14')]
Solution with Recursive Backtracking solution with Forward check = False : took 0.2713801860809326 sec and 264160 checks
1 4 3 2
3 2 4 1
4 1 2 3
2 3 1 4

Solution with Recursive Backtracking solution with Forward check = True : took 0.12480616569519043 sec and 56928 checks
1 4 3 2
3 2 4 1
4 1 2 3
2 3 1 4

Solution with Minimum Conflicts solution : took 0.0009980201721191406 sec and 1430 checks
1 3 2 4
3 1 4 2
```