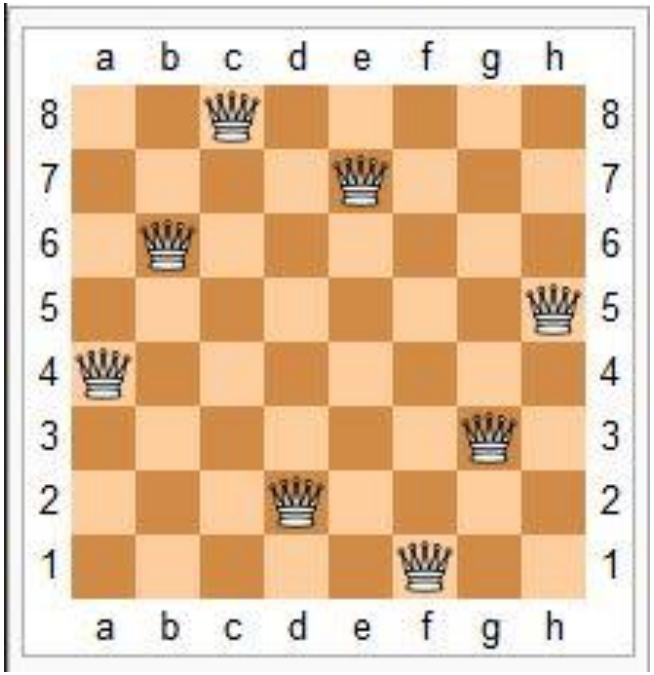
Problem N hetmanów

Szachownica 8x8



Wszystkich ustawień:

$\binom{64}{8}$ = 4 426 165 368

Po jednym w każdym wierszu:

8^8 = 16 777 216

Po jednym w każdym wierszu i w kolumnie:

8! = 40 320

Algorytm z powrotami:

- Podstawą algorytmu jest funkcja test(c), której zadaniem jest wypróbowanie wszystkich możliwych ustawień hetmana w kolumnie o numerze c.
- Dla każdego takiego ustawienia wywoływana jest funkcja test(c+1), próbująca ustawić hetmana w następnej kolumnie.
- Jeśli wyczerpią się wszystkie możliwości w danej kolumnie, wtedy funkcja test kończy działanie i wracamy do funkcji test z poprzedniej kolumny.
- Jeśli uda się ustawić hetmana w ostatniej kolumnie, wypisywana jest konfiguracja hetmanów, czyli kolejne rozwiązanie problemu i wracamy do przedostatniej kolumny.
- Jeśli funkcja test z pierwszej kolumny zakończy działanie, będzie to również koniec całego algorytmu.

Efektywność algorytmu

- Zależy od tego jak szybko uda nam się sprawdzić, czy w danym miejscu możemy postawić hetmana,
- Możemy sprawdzać to w czasie stałym, jeśli będziemy pamiętać, które wiersze i ukośne rzędy są szachowane przez dotychczas ustawione hetmany,
- W tym celu będziemy potrzebować trzy tablice: row, down, up

Organizacja tablic

Na przykład dla N = 4 mamy numerację wierszy i kolumn:

	1	2	3	4
1				
2				
3				
4				

oraz przyjmujemy numerację rzędów ukośnych w jedną i drugą stronę:

up

1	2	3	4
2	3	4	5
3	4	5	6
4	5	6	7

down

4	3	2	1
5	4	3	2
6	5	4	3
7	6	5	4

- Pole szachownicy w wierszu r i kolumnie c należy do rzędu ukośnego up[r+c-1] oraz do rzędu down[r-c+N]

Potrzebne funkcje

- Funkcja possible sprawdza, czy na polu (r,c) można postawić hetmana:

```
possible(r,c)
    return !row[r] && !up[r+c-1] && !down[r-c+N]
```

Potrzebne funkcje

- Numery wierszy, w których stoją hetmany w poszczególnych kolumnach przechowamy w tablicy solution o rozmiarze N. Funkcja put ustawia hetmana na polu (r,c):

```
put(r,c)
    solution[c] = r
    row[r] = up[r+c-1] = down[r-c+N] = true
```

Potrzebne funkcje

- Z kolei funkcja remove usuwa hetmana:

```
remove(r,c)
    row[r] = up[r+c-1] = down[r-c+N] = false
```

Potrzebne funkcje

- Główną funkcją jest test(c):

```
test(c)
   for r=1 to N do
      if possible(r,c)
         put(r,c)
         if c==N
            print solution
         else
            test(c+1)
         remove(r,c)
```