МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”
Інститут прикладної математики і фундаментальних наук

Кафедра прикладної математики

З В І Т

про виконання лабораторної роботи №1

із дисципліни
“Математичні основи штучного інтелекту”

Виконав: студент групи ПМ-32,
Шеремета Данило
Прийняв: доц. каф. Пабирівський
В.В.

Львів — 2023

Лабораторна робота №1

Тема. Алгоритм відпалу.

Постановка задачі

Розв’язати із використанням алгоритму відпалу задачу розстановки N шахових ферзів на шаховій дошці розміру N×N таким чином, аби жоден ферзь не загрожував будь-якому іншому.

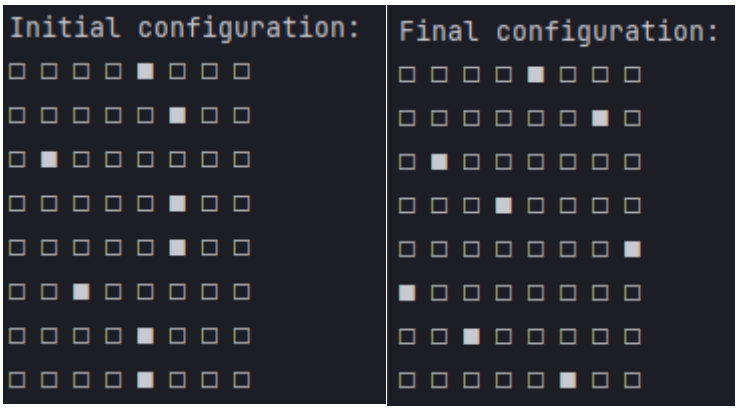
Код програми

```
import random
import math


class Chessboard:
    def __init__(self, n, board=None):
        self.n = n
        if board:
            self.board = board
        else:
            self.board = self.random_board()

    def random_board(self):
        return [random.randint(0, self.n-1) for _ in range(self.n)]

    def print_board(self):
        for row in self.board:
            line = ""
            for col in range(self.n):
                if col == row:
                    line += "■ "
                else:
                    line += "□ "
            print(line)
        print("\n")

    def attacks(self):
        count = 0
        for i in range(self.n):
            for j in range(i + 1, self.n):
                if self.board[i] == self.board[j] or abs(self.board[i] -
self.board[j]) == abs(i - j):
                    count += 1
        return count

    def energy(self):
        return self.attacks()

    def neighbor(self):
        new_board = list(self.board)
        i = random.randint(0, self.n-1)
        j = random.randint(0, self.n-1)
        while new_board[i] == j:
            j = random.randint(0, self.n-1)
        new_board[i] = j
        return new_board
```

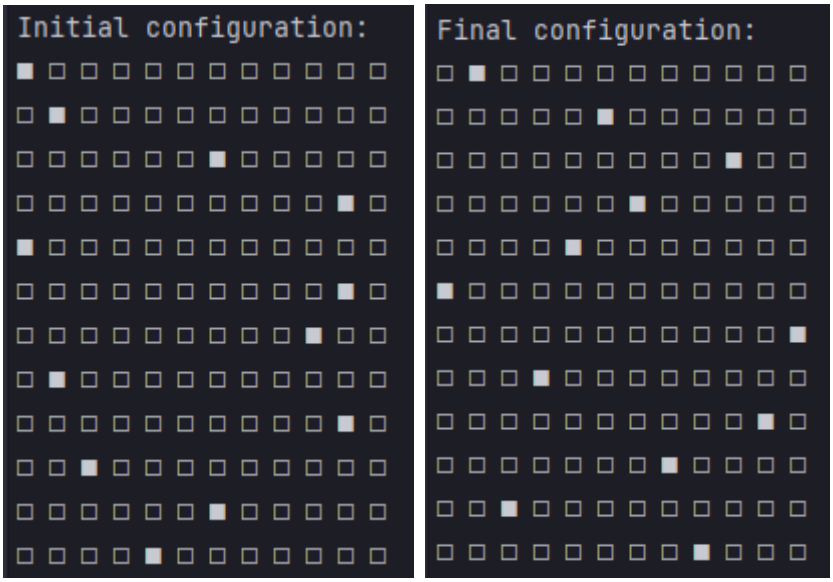
```
class SimulatedAnnealing:
    def __init__(self, temperature=10.0, cooling_rate=0.9999):
        self.temperature = temperature
        self.cooling_rate = cooling_rate

    def acceptance_probability(self, old_energy, new_energy):
        if new_energy < old_energy:
            return 1.0
        else:
            return math.exp((old_energy - new_energy) / self.temperature)

    def anneal(self, chessboard):
        old_energy = chessboard.energy()
        while self.temperature > 0.001 and old_energy > 0:
            new_board = chessboard.neighbor()
            new_energy = Chessboard(chessboard.n, board=new_board).energy()
            if self.acceptance_probability(old_energy, new_energy) >
random.random():
                chessboard.board = new_board[:]
                old_energy = new_energy
            self.temperature *= self.cooling_rate
        return chessboard.board


n = 20 # size of chessboard (n x n)

chessboard = Chessboard(n) # generate initial configuration

print("Initial configuration:")
chessboard.print_board()

annealer = SimulatedAnnealing()
board = annealer.anneal(chessboard) # find solution using simulated annealing

print("Final configuration:")
Chessboard(n, board).print_board()
```

Результат виконання програми

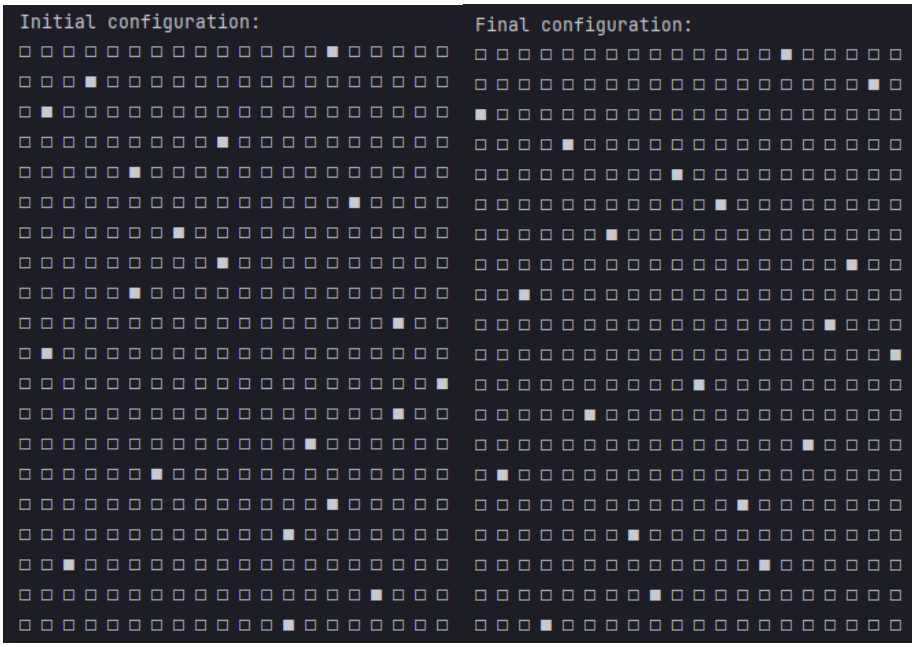
Для n=8:



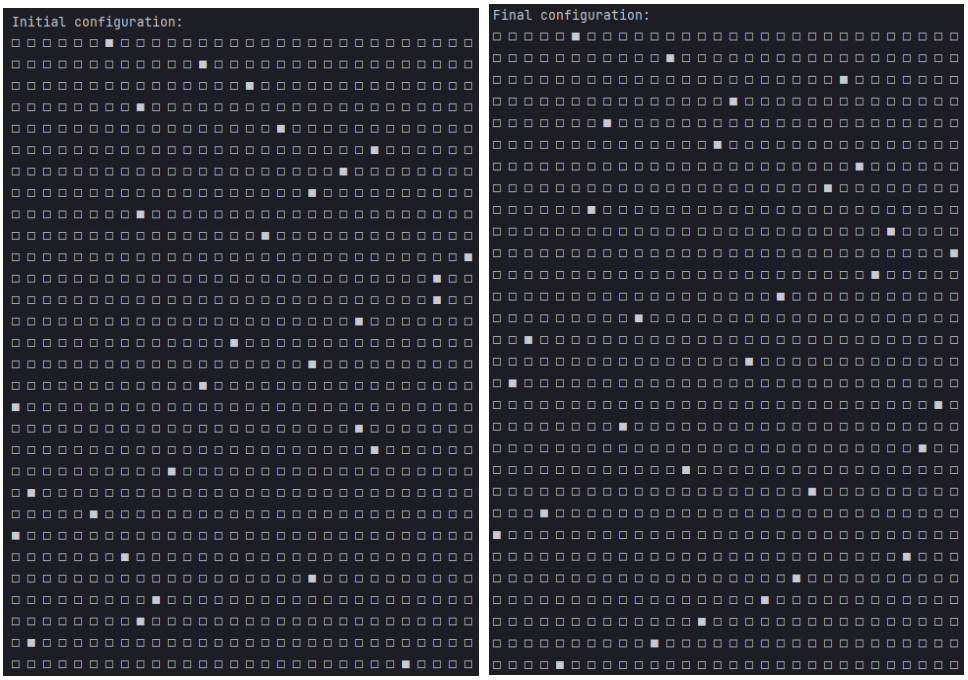
Для n=12:



Для n=20:



Для n=30:



Висновок

У результаті виконання даної лабораторної роботи я навчився використовувати алгоритм відпалу. Створив програму яка розв’язує задачу розстановки N шахових ферзів на шаховій дошці розміру $N \times N$ таким чином, аби жоден ферзь не загрожував будь-якому іншому за допомогою алгоритму відпалу.

Лінк на GitHub: https://github.com/DanSheremeta/mbai_labs/tree/main/Lab1