МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

Лабораторна робота №1

з дисципліни “Методологія інженерії програмного забезпечення”

Виконав

студент групи КП-41мн

Шевченко Гліб Олегович

Київ 2025

**Мета роботи**

* Реалізувати задачу
* Зробити інспекцію коду іншому студенту
* Усунути зауваження, які виникли у результаті інспекції, та навчитись їх
* сортувати та аналізувати

**Хід роботи**

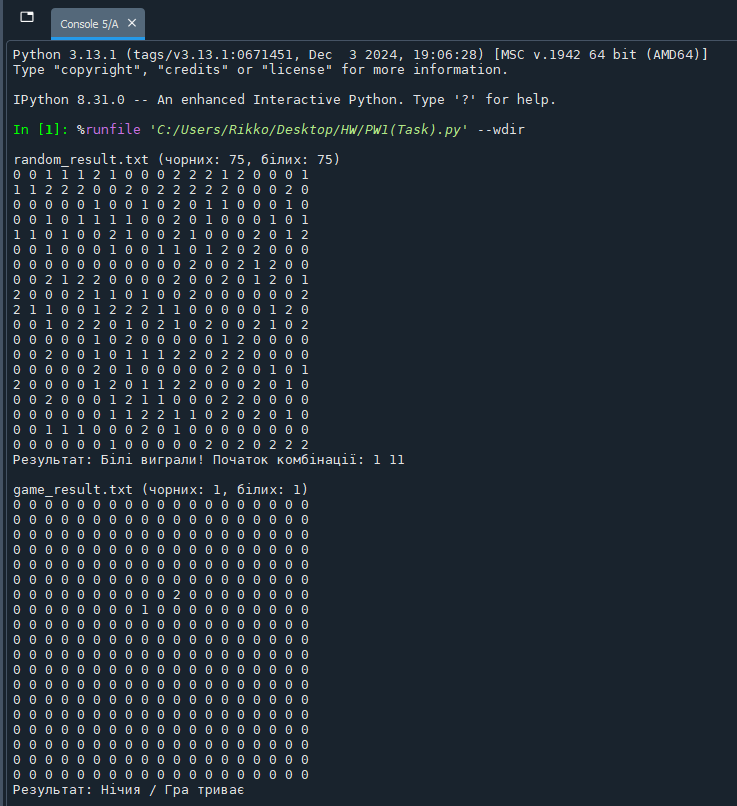


Рис 1. Приклад роботи аналізатору. Код генерує документ з випадковими ситуаціями з гри, а також враховує документ ігрової ситуації.

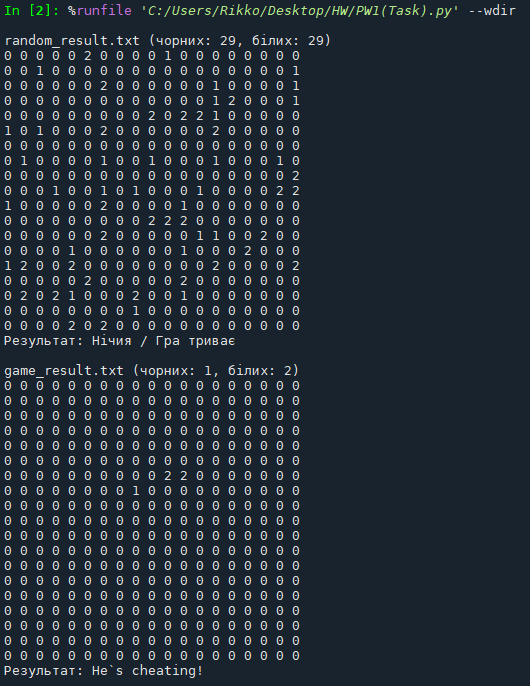


Рис 2. У разі порушення правил програма повідомляє про це.

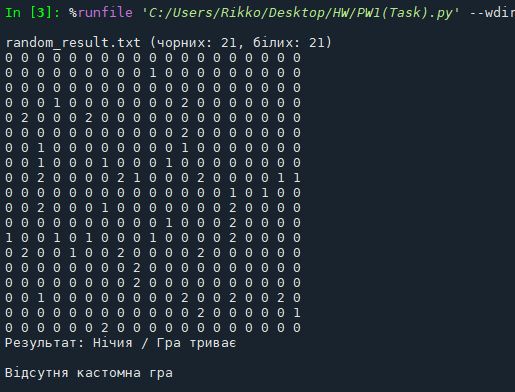


Рис 3. Для роботи програми не обов'язково завантажувати власну гру.

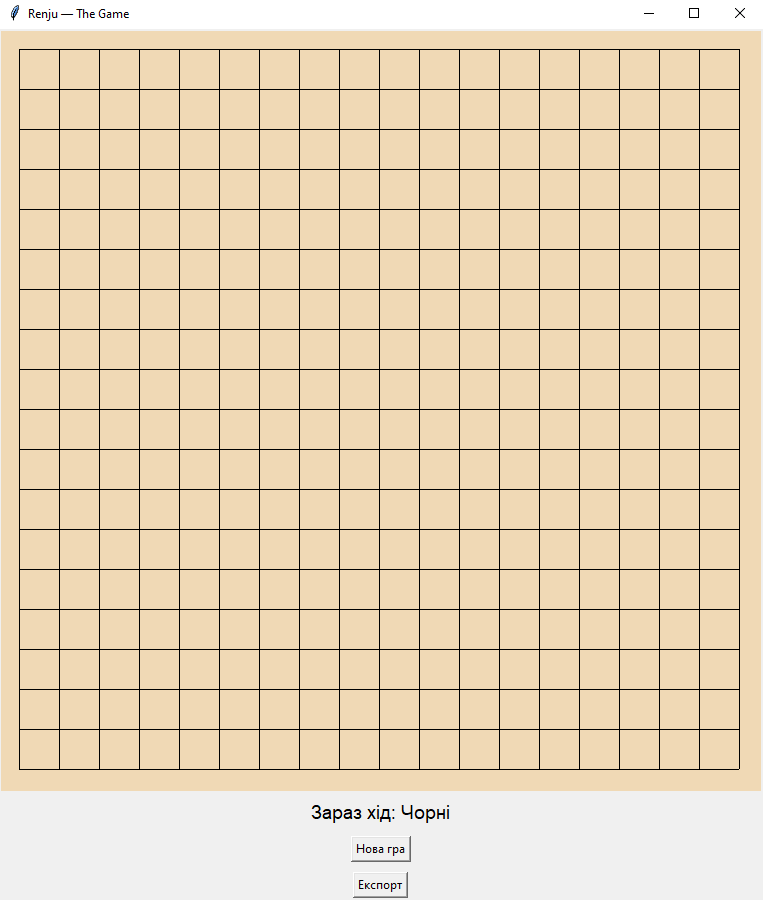


Рис 4. Додатково була реалізована власна гра з можливістю експорту файлу для аналізу. Експорт можна здійснити в будь який момент.

**Код програми**

**Task**

import random

import os

# Генерація random\_result.txt

def generate\_random\_game(filename):

board\_size = 19

board = [[0 for \_ in range(board\_size)] for \_ in range(board\_size)]

total\_cells = board\_size \* board\_size

total\_moves = random.randint(30, 150)

black\_moves = total\_moves // 2 + total\_moves % 2

white\_moves = total\_moves // 2

moves = [1] \* black\_moves + [2] \* white\_moves + [0] \* (total\_cells - total\_moves)

random.shuffle(moves)

idx = 0

for i in range(board\_size):

for j in range(board\_size):

board[i][j] = moves[idx]

idx += 1

with open(filename, 'w') as f:

f.write("1\n")

for row in board:

f.write(" ".join(map(str, row)) + "\n")

return board

# Перевірка перемоги

def check\_win(board, color, x, y):

directions = [(0, 1), (1, 0), (1, 1), (1, -1)]

for dx, dy in directions:

count = 1

nx, ny = x, y

while True:

nx += dx

ny += dy

if 0 <= nx < 19 and 0 <= ny < 19 and board[nx][ny] == color:

count += 1

else:

break

nx, ny = x, y

while True:

nx -= dx

ny -= dy

if 0 <= nx < 19 and 0 <= ny < 19 and board[nx][ny] == color:

count += 1

else:

break

if count == 5:

nx, ny = x, y

while True:

px = nx - dx

py = ny - dy

if 0 <= px < 19 and 0 <= py < 19 and board[px][py] == color:

nx, ny = px, py

else:

break

return (color, nx+1, ny+1)

return (0, -1, -1)

# Аналіз результата

def analyze\_game(board, title=""):

black\_count = sum(row.count(1) for row in board)

white\_count = sum(row.count(2) for row in board)

print(f"\n{title} (чорних: {black\_count}, білих: {white\_count})")

for row in board:

print(" ".join(map(str, row)))

# Перевірка кількості каменів (для кастомної гри)

if title == "game\_result.txt":

if white\_count > black\_count or black\_count - white\_count > 1:

print("Результат: He`s cheating!")

return

result = 0

res\_x, res\_y = -1, -1

for i in range(19):

for j in range(19):

if board[i][j] != 0:

winner, x, y = check\_win(board, board[i][j], i, j)

if winner != 0:

result = winner

res\_x, res\_y = x, y

break

if result != 0:

break

if result == 0:

print("Результат: Нічия / Гра триває")

else:

player = "Чорні" if result == 1 else "Білі"

print(f"Результат: {player} виграли! Початок комбінації: {res\_x} {res\_y}")

# Зчитування поля з файлу

def load\_game\_file(filename):

with open(filename, 'r') as f:

data = f.read().splitlines()

board = []

for line in data[1:]:

board.append(list(map(int, line.split())))

return board

# Основний код

if \_\_name\_\_ == "\_\_main\_\_":

# Генерация и анализ random\_result.txt

random\_board = generate\_random\_game("random\_result.txt")

analyze\_game(random\_board, "random\_result.txt")

# Перевірка наявності game\_result.txt

if os.path.exists("game\_result.txt"):

custom\_board = load\_game\_file("game\_result.txt")

analyze\_game(custom\_board, "game\_result.txt")

else:

print("\nВідсутня кастомна гра")

**Game**

import tkinter as tk

from tkinter import messagebox

# Розмір дошки

BOARD\_SIZE = 19

CELL\_SIZE = 40 # Розмір клітинки в пікселях

STONE\_RADIUS = 16 # Радіус каменю

# Створюємо основне вікно

root = tk.Tk()

root.title("Renju — The Game")

# Ініціалізуємо змінні

board = [[0 for \_ in range(BOARD\_SIZE)] for \_ in range(BOARD\_SIZE)] # 0 — порожньо

current\_player = 1 # 1 — чорний, 2 — білий

# Функція для перевірки перемоги

def check\_win(x, y):

color = board[y][x]

directions = [(0, 1), (1, 0), (1, 1), (1, -1)]

for dx, dy in directions:

count = 1

# Рух вперед

nx, ny = x + dx, y + dy

while 0 <= nx < BOARD\_SIZE and 0 <= ny < BOARD\_SIZE and board[ny][nx] == color:

count += 1

nx += dx

ny += dy

# Рух назад

nx, ny = x - dx, y - dy

while 0 <= nx < BOARD\_SIZE and 0 <= ny < BOARD\_SIZE and board[ny][nx] == color:

count += 1

nx -= dx

ny -= dy

# Перемога тільки при рівно 5 каменях

if count == 5:

return True

return False

# Функція для обробки кліку мишкою

def click(event):

global current\_player

x = event.x // CELL\_SIZE

y = event.y // CELL\_SIZE

# Якщо клітинка порожня

if 0 <= x < BOARD\_SIZE and 0 <= y < BOARD\_SIZE and board[y][x] == 0:

board[y][x] = current\_player

draw\_stone(x, y, current\_player)

if check\_win(x, y):

winner = "Чорні" if current\_player == 1 else "Білі"

messagebox.showinfo("Перемога!", f"{winner} перемогли!")

reset\_board()

return

# Змінюємо гравця

current\_player = 2 if current\_player == 1 else 1

player\_turn\_label.config(text=f"Зараз хід: {'Чорні' if current\_player == 1 else 'Білі'}")

# Функція для малювання каменя

def draw\_stone(x, y, player):

cx = x \* CELL\_SIZE + CELL\_SIZE // 2

cy = y \* CELL\_SIZE + CELL\_SIZE // 2

color = "black" if player == 1 else "white"

canvas.create\_oval(cx - STONE\_RADIUS, cy - STONE\_RADIUS, cx + STONE\_RADIUS, cy + STONE\_RADIUS, fill=color)

# Функція для скидання гри

def reset\_board():

global board, current\_player

board = [[0 for \_ in range(BOARD\_SIZE)] for \_ in range(BOARD\_SIZE)]

current\_player = 1

canvas.delete("all")

draw\_board()

player\_turn\_label.config(text="Зараз хід: Чорні")

# Функція для малювання дошки

def draw\_board():

for i in range(BOARD\_SIZE):

# Горизонтальні лінії

canvas.create\_line(CELL\_SIZE // 2, CELL\_SIZE // 2 + i \* CELL\_SIZE,

CELL\_SIZE // 2 + (BOARD\_SIZE - 1) \* CELL\_SIZE, CELL\_SIZE // 2 + i \* CELL\_SIZE)

# Вертикальні лінії

canvas.create\_line(CELL\_SIZE // 2 + i \* CELL\_SIZE, CELL\_SIZE // 2,

CELL\_SIZE // 2 + i \* CELL\_SIZE, CELL\_SIZE // 2 + (BOARD\_SIZE - 1) \* CELL\_SIZE)

# Створюємо Canvas для гри

canvas = tk.Canvas(root, width=BOARD\_SIZE \* CELL\_SIZE, height=BOARD\_SIZE \* CELL\_SIZE, bg="#F0D9B5")

canvas.pack()

# Малюємо початкову дошку

draw\_board()

# Підпис ходу гравця

player\_turn\_label = tk.Label(root, text="Зараз хід: Чорні", font=("Arial", 14))

player\_turn\_label.pack(pady=5)

# Кнопка для нової гри

reset\_button = tk.Button(root, text="Нова гра", command=reset\_board)

reset\_button.pack(pady=5)

# Прив'язуємо кліки

canvas.bind("<Button-1>", click)

# Функція експорту

def export\_board():

try:

with open("game\_result.txt", "w") as f:

f.write(f"{current\_player}\n") # Текущий игрок первым в файле

for row in board:

f.write(" ".join(str(cell) for cell in row) + "\n")

messagebox.showinfo("Експорт", "Наявна ситуація збережена в game\_result.txt")

except Exception as e:

messagebox.showerror("Помилка", f"Не вдалось зберегти файл:\n{e}")

# Кнопка для експорту

export\_button = tk.Button(root, text="Експорт", command=export\_board)

export\_button.pack(pady=5)

# Запускаємо головний цикл

root.mainloop()

**Висновки**

В ході виконання цієї лабораторної роботи ми створили програму для аналізу ігрової ситуації в грі Рендзю. Додатково було реалізовано гру для створення ігрових ситуацій, для подальшого аналізу.