Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Лабораторна робота №3

з дисципліни
«Алгоритми і структури даних»

Виконав:

студент групи ІМ-43
Калєнченко Михайло Олександрович
номер у списку групи: 12

Перевірив:

Сергієнко А.М.

Київ 2025

Постановка задачі

1. Представити у програмі напрямлений і ненапрямлений графи з заданими параметрами:

- кількість вершин n;
- розміщення вершин;
- матриця суміжності A.

2. Створити програму для формування зображення напрямленого і ненапрямленого графів у графічному вікні.

Завдання за варіантом:

Номер варіанту: 4312
Кількість вершин n = 10
Розміщення вершин - прямокутником
Коефіцієнт k = 0.72

Текст програми:

```
import tkinter as tk
import random
import math

VARIANT = 4312
n3 = (VARIANT // 10) % 10
n4 = VARIANT % 10
n = 10 * n3
random.seed(VARIANT)

def generate_directed_matrix(n, n3, n4):
    T = [[random.uniform(0, 2.0) for _ in range(n)] for _ in range(n)]
    k = 1.0 - n3 * 0.02 - n4 * 0.005 - 0.25
```

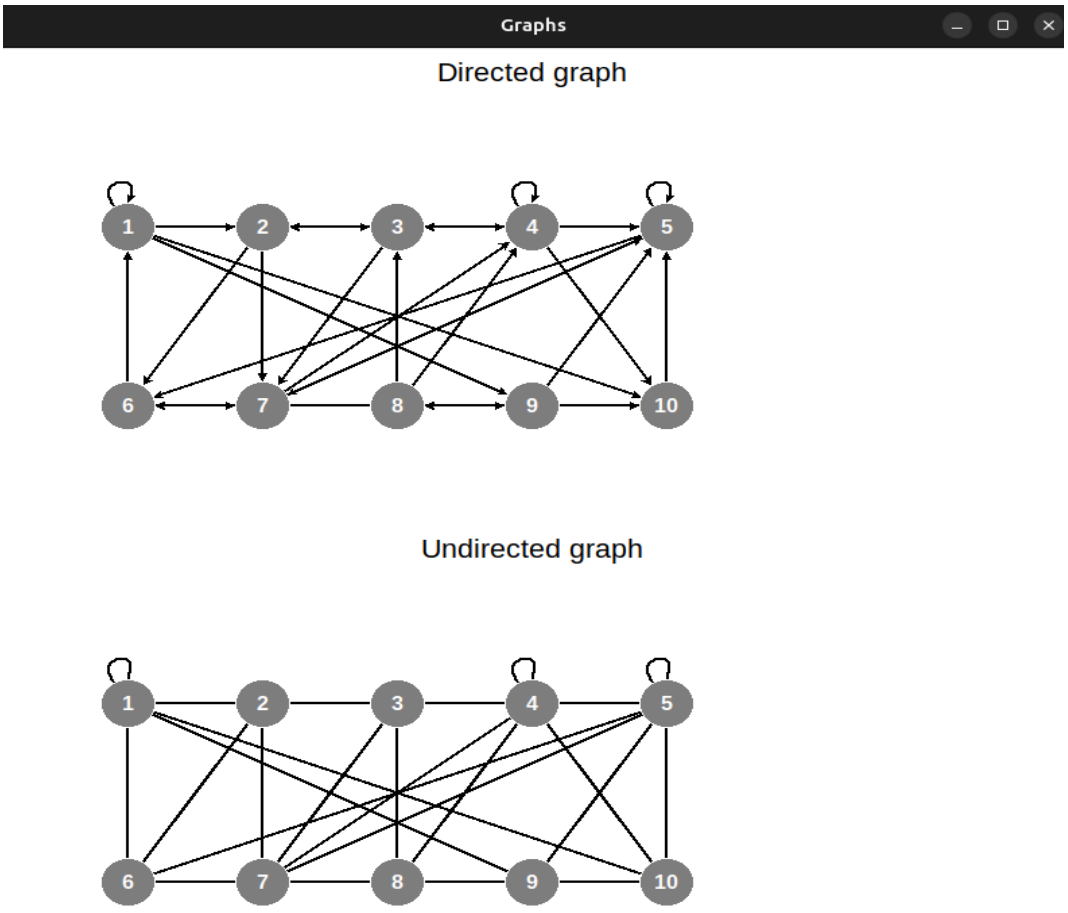
```
    return [[1 if T[i][j] * k >= 1.0 else 0 for j in range(n)] for i in
range(n)]

def generate_undirected_matrix(A_dir):
    return [[1 if A_dir[i][j] or A_dir[j][i] else 0 for j in range(n)]
for i in range(n)]

def calculate_positions(n):
    rows = 2
    cols = 5
    pad_x = 100
    pad_y = 150
    pos = []

    for idx in range(n):
        row = idx // cols
        col = idx % cols
        x = pad_x * (col + 1)
        y = pad_y * (row + 1)
        pos.append((x, y))

    return pos[:n]

def draw_graph(canvas, A, directed, title, y_offset):
    RADIUS = 20
    ARROW_SHAPE = (6, 8, 3)
    LOOP_SIZE = 40

    n = len(A)
    canvas.create_text(400, 20 + y_offset, text=title, font=("Arial",
16))
    pos = calculate_positions(n)

    for i in range(n):
        for j in range(n):
            if A[i][j]:
                x1, y1 = pos[i]
                x2, y2 = pos[j]
                y1 += y_offset
                y2 += y_offset

                if i == j:
                    row = i // 5
```

```
                    base_angle = math.radians(240) if row == 0 else
math.radians(120)

                    start_x = x1 + RADIUS * math.cos(base_angle)
                    start_y = y1 + RADIUS * math.sin(base_angle)
                    ctrl1_x = x1 + LOOP_SIZE * math.cos(base_angle)
                    ctrl1_y = y1 + LOOP_SIZE * math.sin(base_angle)
                    ctrl2_x = x1 + LOOP_SIZE * math.cos(base_angle +
0.7)
                    ctrl2_y = y1 + LOOP_SIZE * math.sin(base_angle +
0.7)
                    end_x = x1 + RADIUS * math.cos(base_angle + 0.5)
                    end_y = y1 + RADIUS * math.sin(base_angle + 0.5)

                    canvas.create_line(
                        start_x, start_y,
                        ctrl1_x, ctrl1_y,
                        ctrl2_x, ctrl2_y,
                        end_x, end_y,
                        smooth=True,
                        arrow=tk.LAST if directed else None,
                        arrowshape=ARROW_SHAPE,
                        width=2
                    )
                else:
                    dx = x2 - x1
                    dy = y2 - y1
                    length = math.hypot(dx, dy)
                    if length == 0: continue
                    ux = dx / length
                    uy = dy / length
                    x_end = x2 - ux * RADIUS
                    y_end = y2 - uy * RADIUS
                    canvas.create_line(
                        x1, y1, x_end, y_end,
                        arrow=tk.LAST if directed else None,
                        arrowshape=ARROW_SHAPE,
                        width=2
                    )

    for idx, (x, y) in enumerate(pos):
        y += y_offset
        canvas.create_oval(
```

```
            x - RADIUS, y - RADIUS,
            x + RADIUS, y + RADIUS,
            fill="grey", outline="white"
        )
        canvas.create_text(x, y, text=str(idx+1), fill="white" ,
font=("Arial", 12, "bold"))

def main():
    A_dir = generate_directed_matrix(n, n3, n4)
    A_undir = generate_undirected_matrix(A_dir)
    print("Undirected graph")
    for row in A_undir:
        print(" ".join(str(val) for val in row))
    print("Directed graph")
    for row in A_dir:
        print(" ".join(str(val) for val in row))

    root = tk.Tk()
    root.title("Graphs")
    canvas = tk.Canvas(root, width=800, height=800, bg="white")
    canvas.pack()

    draw_graph(canvas, A_dir, True, "Directed graph", 0)
    draw_graph(canvas, A_undir, False, "Undirected graph", 400)

    root.mainloop()

if __name__ == "__main__":
    main()
```

Тестування програми:

```
Lab3/main.py
Undirected graph
1 1 1 0 0 1 0 0 1 1
1 0 1 1 1 1 1 0 0 0
1 1 0 0 1 0 1 1 0 0
0 1 0 1 1 0 1 1 0 1
0 1 1 1 1 1 1 0 1 1
1 1 0 0 1 0 1 0 1 0
0 1 1 1 1 1 0 0 0 1
0 0 1 1 0 0 0 0 1 1
1 0 0 0 1 1 0 1 0 0
1 0 0 1 1 0 1 1 0 0
Directed graph
1 1 1 0 0 0 0 0 1 1
0 0 0 1 1 1 1 0 0 0
0 1 0 0 1 0 1 0 0 0
0 0 0 1 1 0 0 0 0 1
0 1 1 0 1 1 1 0 0 0
1 0 0 0 0 0 1 0 0 0
0 0 0 1 1 1 0 0 0 1
0 0 1 1 0 0 0 0 1 0
0 0 0 0 1 1 0 0 0 0
0 0 0 0 1 0 0 1 0 0
```



Висновок

У ході виконання лабораторної роботи №3 «Графічне представлення графів» було досягнуто поставлену мету — набуто практичних навичок представлення графів у комп’ютері та закріплено розуміння принципів роботи операційної системи при побудові графічних додатків.

Було реалізовано програму мовою Python із використанням бібліотеки tkinter, яка:

- генерує матриці суміжності для напрямленого та ненапрямленого графів на основі заданих правил;
- автоматично розташовує вершини графа у вигляді прямокутної сітки (2x5), відповідно до варіанту завдання;
- будує графічне зображення графів, використовуючи базові графічні примітиви (еліпси для вершин, лінії зі стрілками для ребер, дуги для петель);
- виводить матриці суміжності у консоль у зручному форматі.

Особливу увагу приділено:

- дотриманню вимог щодо самостійного формування графічних об'єктів без використання готових бібліотек для роботи з графами;
- універсальності побудови графа через використання циклів для створення вершин і ребер;
- правильному використанню генератора випадкових чисел із фіксованим параметром seed, що забезпечує відтворюваність результатів.

У результаті роботи було отримано повноцінне графічне зображення як напрямленого, так і ненапрямленого графів, що свідчить про успішне виконання лабораторної роботи.