Міністерство освіти і науки України

Кременчуцький національний університет   
імені Михайла Остроградського

Навчально-науковий інститут електричної інженерії   
та інформаційних технологій

Кафедра автоматизації та інформаційних систем

НаВчальна дисципліна  
«**АЛГОРИТМИ І СТРУКТУРИ ДАНИХ**»

Звіт

З ЛАБОРАТОРНОЇ РОБОТИ № 1

Виконав

студент групи КН-24-1

Шпак А.П.

Перевірив

доцент кафедри КІЕ

Сидоренко В.М.

Кременчук 2025

**Тема.** Графи. Ациклічні графи

**Мета:** набути практичних навичок розв’язання задач топографічного сортування та оцінювання їх асимптотичної складності.

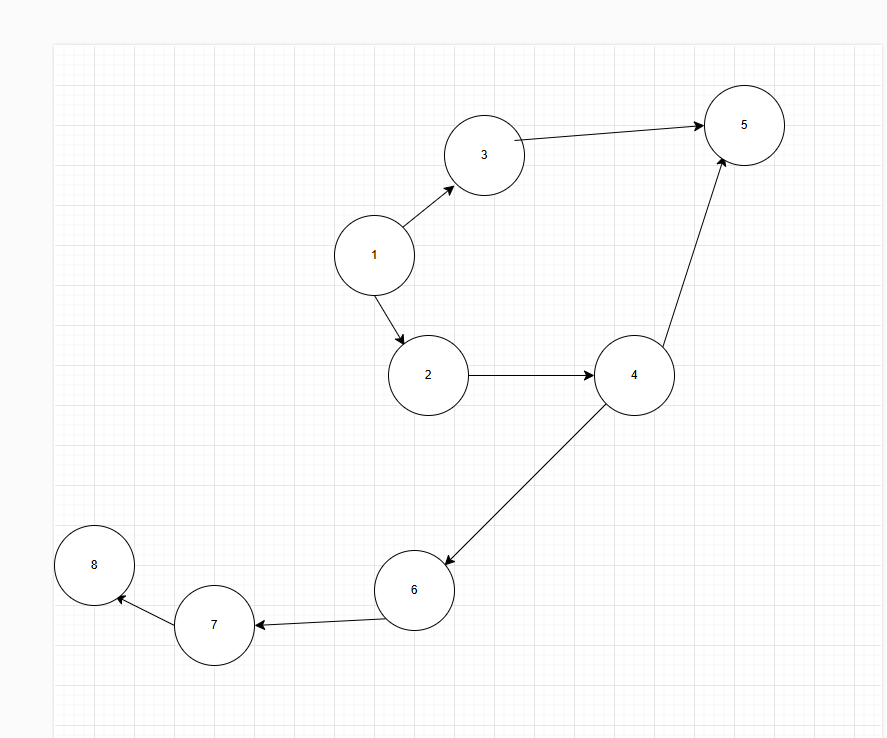
**Хід роботи**

**Завдання №7**

Побудуємо граф за заданими даними:

Вершини: {1,2,3,4,5,6,7,8}

Ребра: {(1,2),(1,3),(2,4),(3,5),(4,5),(4,6),(6,7),(7,8)}

****

**Рисунок 1 – Схема наданого акциклічного графу**

1. **бчислити вхідний степінь** для кожної вершини (кількість вхідних ребер):
   * 1: 0 (немає вхідних ребер)
   * 2: 1 (від 1)
   * 3: 1 (від 1)
   * 4: 1 (від 2)
   * 5: 2 (від 3 і 4)
   * 6: 1 (від 4)
   * 7: 1 (від 6)
   * 8: 1 (від 7)
2. **Почати з вершин, у яких вхідна степінь= 0**:
   * Єдина вершина з вхідною степінь = 0 — це 1. Додаємо її до результату: [1].
3. **Видалити вершину 1 і оновити вхідною степінь**:
   * Видаляємо ребра 1→2 і 1→3.
   * 2: вхідна степінь стає 0.
   * 3: вхідна степінь стає 0.
   * Додаємо вершини 2 і 3 (вхідна степінь = 0) до черги. Черга: [2, 3].
4. **Обробити вершину 2**:
   * Результат: [1, 2].
   * Видаляємо ребро 2→4.
   * 4: вхідна степінь стає 0.
   * Додаємо 4 до черги. Черга: [3, 4].
5. **Обробити вершину 3**:
   * Результат: [1, 2, 3].
   * Видаляємо ребро 3→5.
   * 5: вхідна степінь стає 1 (залишилось ребро від 4).
   * Черга: [4].
6. **Обробити вершину 4**:
   * Результат: [1, 2, 3, 4].
   * Видаляємо ребра 4→5 і 4→6.
   * 5: вхідна степінь стає 0.
   * 6: вхідна степінь стає 0.
   * Додаємо 5 і 6 до черги. Черга: [5, 6].
7. **Обробити вершину 5**:
   * Результат: [1, 2, 3, 4, 5].
   * У 5 немає вихідних ребер.
   * Черга: [6].
8. **Обробити вершину 6**:
   * Результат: [1, 2, 3, 4, 5, 6].
   * Видаляємо ребро 6→7.
   * 7: вхідна степінь стає 0.
   * Додаємо 7 до черги. Черга: [7].
9. **Обробити вершину 7**:
   * Результат: [1, 2, 3, 4, 5, 6, 7].
   * Видаляємо ребро 7→8.
   * 8: вхідна степінь стає 0.
   * Додаємо 8 до черги. Черга: [8].i
10. **Обробити вершину 8**:
    * Результат: [1, 2, 3, 4, 5, 6, 7, 8].
    * У 8 немає вихідних ребер.
    * Черга порожня.

**3. Результат топологічного сортування**

Остаточний порядок: [1, 2, 3, 4, 5, 6, 7, 8].

**Відповідь:**

Топологічне сортування графа за допомогою алгоритму Кана: **1, 2, 3, 4, 5, 6, 7, 8**.

**Висновок:**

Набув практичних навичок розв’язання задач топографічного сортування та оцінювання їх асимптотичної складності.

# **Контрольні питання:**

**1.Які переваги і недоліки алгоритму Кана порівняно з алгоритмом DFS для топологічного сортування графа?**

 **Алгоритм Кана**: Переваги:

* Легше виявляти цикли, оскільки їх наявність можна визначити під час виконання.
* Підходить для густо зв'язаних графів.
* Недоліки:
* Потребує явного зберігання графа і вхідного ступеня для всіх вершин.

 **Алгоритм DFS**: Переваги:

* Ефективний для розріджених графів.
* Легше реалізується рекурсивно.
* Недоліки:
* Виявлення циклів вимагає додаткових перевірок.

**2. Яка складність часу і пам’яті для кожного з алгоритмів у найгіршому і найкращому випадках?**

Обидва алгоритми мають часову складність у найгіршому випадку 𝑂(|𝑉| + | 𝐸|)

Пам’ять:

* **Кан**: додатково зберігає вхідний ступінь вершин.
* **DFS**: зберігає стан кожної вершини та стек викликів (у випадку рекурсії).

**3. Чи можна застосовувати алгоритм Кана до графів з вагами на ребрах? Як це порівняти з DFS?**

Алгоритм Кана не враховує ваги ребер, оскільки призначений для топологічного сортування. Для роботи з вагами краще підходять алгоритми, такі як Дейкстра чи Беллман-Форд. DFS теж можна використовувати для обходу графа з вагами, але додаткову обробку потрібно реалізувати вручну.

**4. Як впливає структура графа на швидкість роботи кожного з цих алгоритмів?**

 Алгоритм Кана краще працює з густо зв'язаними графами завдяки ефективності видалення ребер.

 Алгоритм DFS швидше обробляє розріджені графи, оскільки кількість рекурсивних викликів зменшується.

**5. Чи є обмеження використання кожного алгоритму для певних типів графів або завдань?**

 Алгоритм Кана не працює для графів з циклами.

 DFS потребує уваги для графів великої глибини, оскільки може статися переповнення стека.

**6. Які варіанти оптимізації можна застосувати для кожного алгоритму з метою поліпшення його продуктивності?**

 **Для алгоритму Кана**:

* Використання черг із пріоритетами для ефективнішого вибору вершин.

 **Для DFS**:

* Використання ітеративного підходу замість рекурсивного, щоб уникнути переповнення стека.