**ПОСТАНОВКА ЗАДАЧІ**

**КУРСОВОЇ РОБОТИ**

**НА ТЕМУ: «Генерація перестановок Хіпа»**

***Латко Артем Андрійович, група КМ-32***

**Вступ**

Перестановки, або розміщення, є однією з фундаментальних комбінаторних задач, яка має широке застосування в різних областях, від алгоритмів до криптографії. Одним із способів ефективного генерування всіх можливих перестановок елементів масиву є використання алгоритму перестановок Хіпа, який отримав свою назву від внесення певних оптимізацій у класичний алгоритм генерації перестановок.

**Мета:** дослідити алгоритм генерації перестановок Хіпа та реалізувати його на мові програмування C. Основною метою є вивчення теоретичних основ алгоритму, розкриття його ефективності (проведення експериментальних тестів з різними розмірами вхідних даних для оцінки швидкодії алгоритму) та можливості використання у практичних задачах програмування.

**Постановка задачі**

Моя програма має 3 опції: 1) Розв'язання так званої задачі комівояжера; 2) Створення розкладу маршрутів між містами і пошук найоптимальнішого за часом та вартістю; 3) Вихід з програми;

1) Задача комівояжера полягає в тому, що рахується найкоротший шлях, який починається з одного міста, а потім потрібно обійти усі інші міста, побувавши в кожному лише по одному разу і повернутися у те місто, з якого починав. Отже задача починається з того, що користувач задає кількість міст, назву кожного міста, та усі можливі відстані між містами. Далі йому пропонується ввести місто з якого починати обхід і після того як користувач вказує його відбувається пошук найкоротшого маршруту саме за допомогою алгоритму перестановок Хіпа.

2) При виклику другої опції, а саме створення розкладу маршрутів між містами і пошук найоптимальнішого за часом та вартістю, користувачу потрібно ввести кількість міст і назву кожного міста. Так як суть цієї опції саме пошук найдешевшого і найшвидшого шляху то цей раз відстані між містами вводити не треба. Тобто це типу як прототип розкладу руху поїздів(електричок), який містить вартість маршруту і час, але тут МІЖ УСІМА МІСТАМИ Є ШЛЯХ, але при цьому і вартість маршруту може бути більшою і час також. (ТАКОЖ СЛІД НЕ ЗАБУВАТИ, ЩО МАРШРУТ ВІД ОДНОГО МІСТА ДО ІНШОГО МОЖЕ БУТИ НЕ ПРОСТО ПРЯМИМ А, НАПРИКЛАД, ЦЕ МОЖЕ БУТИ ЯК ЯКАСЬ КРИВА). Тож після того як користувач ввів назви усіх міст йому треба буде ввести час(в годинах) і вартість(грн) кожного маршруту між кожним містом(АЛЕ СЛІД ВРАХОВУВАТИ ТЕ ЩО ВАРТІСТЬ І ЧАС НАПРИКЛАД ВІД МІСТА 1 ДО МІСТА 2 МОЖУТЬ БУТИ ОДНИМИ А ВІД МІСТА 2 ДО МІСТА 1 МОЖУТЬ БУТИ ІНШИМИ ТАК ЯК ШЛЯХИ МОЖУТЬ БУТИ РІЗНИМИ БО МОЖУТЬ ЇХАТИ ГЕТЬ ПО-ІНШОМУ МАРШРУТУ. ТОЖ ЦЕ І УСКЛАДНЮЄ МОЄ ЗАВДАННЯ, БО ТУТ Я ВЖЕ ЗАДАЮ УЯВНИЙ ГРАФ У ЯКОГО ВІДСТАНІ МІЖ ВЕРШИНАМИ(МІСТАМИ) МАЄ ОРІЄНТОВАНІСТЬ В ОБОХ НАПРЯМКАХ). Тобто користувач ніби заповнує таблицю маршрутів. Так як дані потрібно структуризувати, то для цього в мові програмування С зручно використати структури даних(struct) в яких я буду зберігати саме інформацію про кожен маршрут, так як структура – це набір даних, що може бути різних типів, і для виконання саме цієї курсової роботи як на мене вона підходить ідеально. Після того як всі дані заповнені виводиться таблиця усіх маршрутів та інформацію про них. Після цього користувачу пропонується знайти маршрут який його цікавить. Якщо користувач хоче знайти маршрут, то йому треба вказати початкове місто і кінцеве місто, до якого треба добратися і вже після цього йому виводиться найоптимальніший шлях за часом і вартістю теж у виді таблички, яка може містити декілька шляхів через які буде оптимальніше добратися від того початкового міста до кінцевого. Ну і якщо користувач хоче, то може вивести інформацію про ще один шлях який його цікавить а якщо не захоче то завдання завершується.

**Опис алгоритму роботи користувача з програмою:**

* Вибір однієї з 3 опцій користувачем.
* Перевірка вводу.

Якщо вибрано опцію 1:

* Введення кількості міст
* Введення відстаней (з перевіркою вводу) між кожним містом
* Вивід найкоротшого шляху між містами
* Вивід дистанції шляху
* Завершення опції

Якщо вибрано опцію 2:

* Створення масиву структур розмірність якого залежить від введеної користувачем кількості міст.
* Введення даних (з перевіркою вводу) у відповідні поля структури масиву, що здійснюється в циклі.
* Виведення таблиці із фіксованою шириною кожної комірки відповідно для кожної змінної і інформацією про кожен маршрут.
* Запит про те чи хоче користувач знайти оптимальний шлях. Якщо так то виодиться у вигляді таблички оптимальний шлях, а якщо ні, то опція завершується.

Якщо вибрано опцію 2:

* Вихід з програми

**Опис формату вхідних та вихідних даних:**

Формат вхідних даних:

1)Опція – (int)

2)Кількість міст – (double)

3)Назва міста - (string)

4)Відстань між містами – (int)

5)Час маршруту - (int)

6)Вартість маршруту - (int)

7)Початкове місто - (string)

8)Кінцеве місто - (string)

Формат вихідних даних:

1)Оптимальний шлях - (string)

2)Дистанція шляху - (int)

3)Таблиця з інформацією про кожен маршрут

4)Оптимальний маршрут у вигляді міні-таблички

**ШЛЯХ ВВЕДЕННЯ ВХІДНИХ ДАНИХ**

Введення вхідних даних здійснюється через консоль за допомогою клавіатури.

**ШЛЯХ ВИВЕДЕННЯ ВИХІДНИХ ДАНИХ**

Проміжні результати виводяться у вигляді невеличких текстових повідомлень або таблиці в консолі.