МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФРАСТРУКТУРИ ТА ТЕХНОЛОГІЙ
ІНСТИТУТ УПРАВЛІННЯ, ТЕХНОЛОГІЙ ТА ПРАВА
ФАКУЛЬТЕТ УПРАВЛІННЯ І ТЕХНОЛОГІЙ

Кафедра інформаційних технологій

Лабораторна робота №1

з дисципліни «Фундаментальні комп’ютерні алгоритми»

з теми: «АЛГОРИТМИ ПОБУДОВИ ХЕШ-ТАБЛИЦІ»

Варіант 13

Виконав:
Студент групи
ІПЗд-23121 маг.
Петренко Д.М.
Перевірив:
Доцент кафедри ІТ
Ткаченко О.А.

Київ – 2024

Лабораторна робота №2

Мета: вивчити алгоритми побудови хеш-таблиці, способи вирішення колізій; отримати практичні навички створення і застосування хеш-таблиці шляхом реалізації її на мові java.

Завдання

Завдання першого рівня

Виконайте наступне:

- описати елемент хеш-таблиці, що представляє геометричну фігуру (Вектор: полярні координати, конструктор, методи обчислення координат кінця вектору, виведення об’єкта). Елемент хеш-таблиці визначається як клас. При створенні нового елемента хеш-таблиці координати геометричної фігури повинні бути визначені з використанням генератора випадкових чисел. Необхідно перевірити існування фігури з такими координатами.
- описати хеш-таблицю з відкритою адресацією, яка використовує зазначену у варіанті хеш-функцію (ділення) для вказаного ключа (Координата X). За допомогою хеш-функції обчислити хеш-код для кожної створеної геометричної фігури. Сама хеш таблиця визначаються як клас. Клас хеш-таблиці містить одновимірний масив і розмір хеш таблиці, який задається користувачем виходячи з кількості елементів хеш-таблиці.
- створити екземпляр хеш-таблиці (для наповнення її масивом об’єктів-геометричних фігур) із заданим розміром.
- вставляти елементи у створену хеш-таблицю. Метод вставки повинен повертати логічне значення, яке вказує на успіх операції. Для завдання першого рівня цей метод повертає false, якщо позиція зайнята і тому новий елемент не буде вставлений.
- вивести хеш-таблицю на екран.

13	Вектор: полярні координати, конструктор, методи обчислення координат кінця вектору, виведення об’єкта	Ділення	Координата X

Завдання другого рівня

Виконайте наступне:

- змінити опис хеш-таблиці із завдання першого рівня так, щоб при виникненні колізії її було розв’язано заданим методом (Роздільне зв'язування);
- вставляти елементи в хеш-таблицю, з урахуванням колізії. Метод вставки повинен повертати логічне значення, яке вказує на успіх операції. Для завдання другого рівня цей метод повертає false, якщо колізію не можна вирішити (тільки для хеш-таблиць з відкритою адресацією).
- вивести хеш-таблицю. Метод класу хеш-таблиці, який реалізує операцію виведення, повинен виводити хеш-таблицю, розміщуючи на окремому рядку номер позиції хеш-таблиці, ключ і сам елемент. Якщо елемента в позиції немає, то поряд з номером позиції повинно виводитися відповідне повідомлення «Null». У випадку хеш-таблиці з роздільним зв’язуванням в окремому рядку слід виводити всі елементи, що зберігаються в одній позиції.

Варіант	Метод вирішення колізій
1,5,9,13, 16,20	Роздільне зв'язування

Завдання третього рівня

Виконайте наступне:

- видалити елементи з хеш-таблиці за заданими користувачем критеріями та вивести вміст хеш-таблиці після видалення. Позиції з видаленими елементами позначати як «Deleted»

І3	Елементи зі значенням координати Y меншої від заданої

ХІД РОБОТИ:

Створимо програму за даним завданням на мові програмування Java.

Імпортуємо бібліотеки Java для коректної роботи програми:

```
1  import java.util.Random;
2  import java.util.Scanner;
3  import java.util.ArrayList;
4  import java.util.Iterator;
```

- import java.util.Random;: Цей рядок імпортує клас Random з пакету java.util. Клас Random дозволяє генерувати випадкові значення. В програмі цей клас використовується для генерації випадкових чисел для модуля і кута вектора.
- import java.util.Scanner;: Цей рядок імпортує клас Scanner з пакету java.util. Клас Scanner дозволяє зручно зчитувати введені дані з консолі. В програмі цей клас використовується для зчитування розміру хеш-таблиці та значення координати Y для видалення.
- import java.util.ArrayList;: Цей рядок імпортує клас ArrayList з пакету java.util. Клас ArrayList реалізує динамічний масив, який може змінювати свій розмір під час виконання програми. В програмі цей клас використовується для зберігання списків елементів у хеш-таблиці.
- import java.util.Iterator;: Цей рядок імпортує інтерфейс Iterator з пакету java.util. Інтерфейс Iterator надає методи для ітерації по колекціях, таких як списки або масиви. В програмі цей інтерфейс використовується для безпечного видалення елементів з хеш-таблиці під час її перегляду.

Створюємо клас Vector, який представляє геометричну фігуру вектор, у відповідності з варіантом.

```
class Vector {
    private double magnitude; // Модуль вектора
    private double angle; // Кут вектора з віссю Ox
```

На скріншоті представлено оголошення змінних "Модуль вектора" та "Кут вектора з віссю OX", що доступні лише в межах класу Vector і не можуть бути доступні для зовнішніх класів безпосередньо (private). Це забезпечує ізоляцію та контроль доступу до даних.

```java
// Конструктор, що генерує випадкові полярні координати вектора
public Vector() {
    Random random = new Random();
    this.magnitude = random.nextDouble() * 100; // Генеруємо модуль від 0 до 100
    this.angle = random.nextDouble() * 360; // Генеруємо кут від 0 до 360
}
```

Тут представлено конструктор класу Vector(), що викликається автоматично при створенні класу Vector. В цьому конструкторі створюються випадкові значення для модуля вектора (від 0 до 100), та кут (від 0 до 360).

```java
// Метод для обчислення координат кінця вектора
public double[] calculateEndCoordinates() {
    double[] endCoordinates = new double[2];
    endCoordinates[0] = magnitude * Math.cos(Math.toRadians(angle)); // x = r * cos(θ)
    endCoordinates[1] = magnitude * Math.sin(Math.toRadians(angle)); // y = r * sin(θ)
    return endCoordinates;
}
```

Тут представлено метод calculateEndCoordinates(), що доступний для виклику з будь-якої частини програми, навіть ззовні класу Vector. Метод буде повертати масив дійсних чисел, а саме створює новий масив endCoordinates з двома елементами типу double, який буде містити координати: endCoordinates[0] (знечення координати X) та endCoordinates[1] (знечення координати У).

```java
// Перевизначений метод для виведення даних про вектор
@Override
public String toString() {
    double[] endCoordinates = calculateEndCoordinates();
    return "Magnitude: " + magnitude + ", Angle: " + angle + ", X: " + endCoordinates[0] + ", Y: " + endCoordinates[1];
}
```

Це оголошення методу toString(). public - це модифікатор доступу, що вказує на те, що метод є доступним для виклику з будь-якої частини програми. String - це тип, який вказує на те, що метод повертає рядок. toString() - це ім'я методу. double[] endCoordinates = calculateEndCoordinates() - рядок викликає метод calculateEndCoordinates(), щоб отримати координати кінця вектора і зберігає їх у масиві endCoordinates.

Далі розглянемо клас SeparateChainingHashTable. Цей клас потрібний для вирішення колізій та представлення хеш-таблиці з роздільним зв’язуванням.

```java
// Клас, що представляє хеш-таблицю з роздільним зв'язуванням
class SeparateChainingHashTable {
    private ArrayList<Vector>[] table; // Масив списків елементів хеш-таблиці
```

Тут створено динамічний маси списків елементів хеш-таблиці.

```java
// Конструктор, ініціалізує масив списків заданого розміру
public SeparateChainingHashTable(int size) {
    this.table = new ArrayList[size];
    for (int i = 0; i < size; i++) {
        table[i] = new ArrayList<>();
    }
}
```

Ця частина коду представляє конструктор класу SeparateChainingHashTable, який ініціалізує масив списків заданого розміру **size** для створення хеш-таблиці з роздільним зв'язуванням.

```java
// Метод вставки елементу у хеш-таблицю з урахуванням колізій
public boolean insert(Vector vector) {
    int hash = hashCode(vector); // Отримати хеш-код елементу

    if (table[hash].size() < 5) { // Обмеження на кількість елементів в одній позиції таблиці
        table[hash].add(vector); // Додати елемент у список
        return true; // Успішно вставлено
    }

    return false; // Колізія не вирішена
}

// Метод обчислення хеш-коду елементу
private int hashCode(Vector vector) {
    // Обчислення хеш-коду за допомогою хеш-функції ділення за координатою X
    return Math.abs((int) Math.floor(vector.calculateEndCoordinates()[0])) % table.length;
}
```

Ця частина коду відповідає за вставку елементу у хеш-таблицю з урахуванням колізій, тобто ситуацій, коли два або більше елементів мають однаковий хеш-код і спробують бути поміщеними у ту саму позицію таблиці.

public boolean insert(Vector vector) - це публічний метод, який приймає екземпляр класу Vector як аргумент і повертає значення типу boolean. Він відповідає за вставку нового елементу vector у хеш-таблицю.

private int hashCode(Vector vector) - це приватний метод, який приймає екземпляр класу Vector як аргумент і повертає хеш-код цього вектора. Він використовується для обчислення хеш-коду, який визначає позицію елементу в хеш-таблиці.

Метод виводу хеш-таблиці.

```java
// Метод виведення вмісту хеш-таблиці
public void printTable() {
    for (int i = 0; i < table.length; i++) {
        System.out.print(i + ": ");
        if (table[i].isEmpty()) {
            System.out.println(x:"Null"); // Позиція порожня
        } else {
            for (Vector vector : table[i]) {
                System.out.println(i + ": " + vector); // Вивести елемент
            }
        }
    }
}
```

Видалення елемента з координатою У менше заданої:

```java
// Метод видалення елементів з хеш-таблиці за заданим критерієм
public void removeByYCoordinate(double yCoord) {
    for (int i = 0; i < table.length; i++) {
        if (!table[i].isEmpty()) {
            // Використовуємо ітератор для безпечного видалення елементів
            Iterator<Vector> iterator = table[i].iterator();
            while (iterator.hasNext()) {
                Vector vector = iterator.next();
                if (vector.calculateEndCoordinates()[1] < yCoord) {
                    iterator.remove(); // Видалити елемент
                    System.out.println("Deleted element at position " + i);
                }
            }
        }
    }
}
```

Переходимо до головного класу Main:

```java
public class Main {
    Run | Debug
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print(s:"Введіть розмір хеш-таблиці: ");
        int size = scanner.nextInt();
        SeparateChainingHashTable hashTable = new SeparateChainingHashTable(size); // Створення хеш-таблиці

        // Вставка випадкових векторів у хеш-таблицю
        for (int i = 0; i < size; i++) {
            Vector vector = new Vector(); // Створення нового вектора
            boolean inserted = hashTable.insert(vector); // Вставка в хеш-таблицю
            if (!inserted) {
                System.out.println("Failed to insert vector: " + vector);
            }
        }

        // Виведення вмісту хеш-таблиці
        System.out.println(x:"\nХеш-таблиця:");
        hashTable.printTable();

        // Введення значення координати X
        System.out.print(s:"\nВведіть значення координати Y для видалення: ");
        double xCoord = scanner.nextDouble();

        // Видалення елементів з хеш-таблиці за заданим критерієм (координата X менше введеного значення)
        System.out.println("\nВидалення елементів з координатою Y менше " + xCoord + ":");
        hashTable.removeByYCoordinate(xCoord);

        // Виведення вмісту хеш-таблиці після видалення
        System.out.println(x:"\nХеш-таблиця після видалення:");
        hashTable.printTable();

        scanner.close();
    }
}
```

Цей код представляє використання розробленої раніше хеш-таблиці в контексті основного програмного класу Main. Давайте розберемо його:

1. public class Main { ... }: Це визначення класу Main, який містить метод main. Метод main є входом в програму Java і викликається автоматично при запуску програми.

2. public static void main(String[] args) { ... }: Це метод main, який є головною точкою входу у програму. Він має модифікатори public та static, що означає, що він доступний для виклику з інших класів і є статичним (не потребує створення екземпляра класу для його виклику). Параметр String[] args приймає аргументи командного рядка, передані програмі при її запуску.

3. Створення об'єкта Scanner: Scanner scanner = new Scanner(System.in); - це рядок, який створює об'єкт Scanner, який дозволяє читати введення з консолі.

4. Отримання розміру хеш-таблиці від користувача: int size = scanner.nextInt(); - ця інструкція використовує об'єкт Scanner, щоб отримати ціле число від користувача.

5. Створення хеш-таблиці: SeparateChainingHashTable hashTable = new SeparateChainingHashTable(size); - тут створюється об'єкт SeparateChainingHashTable з переданим розміром, який був отриманий від користувача.

6. Вставка випадкових векторів у хеш-таблицю: for (int i = 0; i < size; i++) { ... } - цикл вставки випадкових векторів у створену хеш-таблицю.

7. Виведення вмісту хеш-таблиці до видалення елементів: hashTable.printTable(); - це виводить вміст хеш-таблиці перед видаленням елементів.

8. Видалення елементів з хеш-таблиці за заданим критерієм: hashTable.removeByYCoordinate(xCoord); - це викликає метод removeByYCoordinate з переданим значенням координати Y, яке отримується також від користувача.

9. Виведення вмісту хеш-таблиці після видалення: hashTable.printTable(); - це виводить вміст хеш-таблиці після виконання операції видалення.

РЕЗУЛЬТАТИ РОБОТИ ПРОГРАМИ:

Приклад 1:

```
Введіть розмір хеш-таблиці: 25
```

```
Хеш-таблиця:
0: 0: Модуль вектора: 44.86431467658385, Кут: 89.61315350437135, X: 0.30291011304754956, Y: 44.86329208677124
1: 1: Модуль вектора: 2.531967765541865, Кут: 260.67415038086807, X: -0.41030293411530794, Y: -2.498502004801964
1: Модуль вектора: 8.779032365347017, Кут: 266.73256061129706, X: -0.5003756119646312, Y: -8.764760893416401
2: 2: Модуль вектора: 87.17831475140757, Кут: 107.70371862729642, X: -26.510480051872474, Y: 83.04970204832019
3: Null
4: 4: Модуль вектора: 33.72739189627776, Кут: 29.18758230331037, X: 29.4449504245676, Y: 16.447852705440273
5: 5: Модуль вектора: 31.21137336249381, Кут: 344.2363326703418, X: 30.037527957991152, Y: -8.479194616582872
5: Модуль вектора: 67.67537376370979, Кут: 324.5816683699874, X: 55.15153250147287, Y: -39.22071744367765
6: 6: Модуль вектора: 17.516110800894857, Кут: 292.6177061508387, X: 6.736356481000972, Y: -16.168971487082956
7: 7: Модуль вектора: 9.166752284582024, Кут: 33.07858806199969, X: 7.6810302025329955, Y: 5.003111279460581
7: Модуль вектора: 8.97178343682028, Кут: 221.49593690115543, X: -6.719890091955389, Y: -5.944407051106317
7: Модуль вектора: 36.60951826864134, Кут: 334.12156993509456, X: 32.93839472130065, Y: -15.978703985172817
7: Модуль вектора: 34.43125211010481, Кут: 100.83374506627055, X: -6.471691688020538, Y: 33.81757425606992
8: 8: Модуль вектора: 46.569994329606224, Кут: 44.21955400813321, X: 33.375440600608265, Y: 32.478367209187496
9: Null
10: Null
11: 11: Модуль вектора: 32.38410269387826, Кут: 109.8514248151873, X: -10.997066606460129, Y: 30.459721491516184
11: Модуль вектора: 87.26285233222426, Кут: 355.40685644440987, X: 86.98260493547507, Y: -6.987977947492654
12: Null
13: 13: Модуль вектора: 38.38276775631333, Кут: 5.338968432478208, X: 38.21624965010343, Y: 3.5714315499608236
14: 14: Модуль вектора: 26.275589916876974, Кут: 56.681724158748075, X: 14.43290277176987, Y: 21.9567288788773
14: Модуль вектора: 70.00782213488651, Кут: 154.2608377654157, X: -63.061674065475614, Y: 30.402638440267417
15: Null
16: Null
17: 17: Модуль вектора: 89.71406437790893, Кут: 40.79152109477885, X: 67.92177743324757, Y: 58.61096738249564
18: 18: Модуль вектора: 33.34437746305491, Кут: 303.7814039968934, X: 18.540336843758027, Y: -27.714678748249508
19: Null
20: 20: Модуль вектора: 26.882408029975934, Кут: 41.15418673396104, X: 20.240876629756333, Y: 17.69098003924877
20: Модуль вектора: 70.15444733176675, Кут: 171.19082433051673, X: -69.3268962639076, Y: 10.743739332235762
20: Модуль вектора: 21.89070046875381, Кут: 23.54154769344462, X: 20.068752415407136, Y: 8.74345146391221
21: 21: Модуль вектора: 95.49414363126671, Кут: 299.1435569477144, X: 46.50559944575634, Y: -83.40480014999055
22: Null
23: 23: Модуль вектора: 76.88229201214963, Кут: 160.95677011530535, X: -72.67472899050657, Y: 25.08526641672102
24: Null
```

```
Введіть значення координати Y для видалення: 0

Видалення елементів з координатою Y менше 0.0:
Видалено елемент на позиції 1
Видалено елемент на позиції 1
Видалено елемент на позиції 5
Видалено елемент на позиції 5
Видалено елемент на позиції 6
Видалено елемент на позиції 7
Видалено елемент на позиції 7
Видалено елемент на позиції 11
Видалено елемент на позиції 18
Видалено елемент на позиції 21
```

```
Хеш-таблиця після видалення:
0: 0: Модуль вектора: 44.86431467658385, Кут: 89.61315350437135, X: 0.30291011304754956, Y: 44.86329208677124
1: Null
2: 2: Модуль вектора: 87.17831475140757, Кут: 107.70371862729642, X: -26.510480051872474, Y: 83.04970204832019
3: Null
4: 4: Модуль вектора: 33.72739189627776, Кут: 29.18758230331037, X: 29.4449504245676, Y: 16.447852705440273
5: Null
6: Null
7: 7: Модуль вектора: 9.166752284582024, Кут: 33.07858806199969, X: 7.6810302025329955, Y: 5.003111279460581
7: Модуль вектора: 34.43125211010481, Кут: 100.83374506627055, X: -6.471691688020538, Y: 33.81757425606992
8: 8: Модуль вектора: 46.569994329606224, Кут: 44.21955400813321, X: 33.375440600608265, Y: 32.478367209187496
9: Null
10: Null
11: 11: Модуль вектора: 32.38410269387826, Кут: 109.8514248151873, X: -10.997066606460129, Y: 30.459721491516184
12: Null
13: 13: Модуль вектора: 38.38276775631333, Кут: 5.338968432478208, X: 38.21624965010343, Y: 3.5714315499608236
14: 14: Модуль вектора: 26.275589916876974, Кут: 56.681724158748075, X: 14.43290277176987, Y: 21.9567288788773
14: Модуль вектора: 70.00782213488651, Кут: 154.2608377654157, X: -63.061674065475614, Y: 30.402638440267417
15: Null
16: Null
17: 17: Модуль вектора: 89.71406437790893, Кут: 40.79152109477885, X: 67.92177743324757, Y: 58.61096738249564
18: Null
19: Null
20: 20: Модуль вектора: 26.882408029975934, Кут: 41.15418673396104, X: 20.240876629756333, Y: 17.69098003924877
20: Модуль вектора: 70.15444733176675, Кут: 171.19082433051673, X: -69.3268962639076, Y: 10.743739332235762
20: Модуль вектора: 21.89070046875381, Кут: 23.54154769344462, X: 20.068752415407136, Y: 8.74345146391221
21: Null
22: Null
23: 23: Модуль вектора: 76.88229201214963, Кут: 160.95677011530535, X: -72.67472899050657, Y: 25.08526641672102
24: Null
```

Приклад 2:

```
Введіть розмір хеш-таблиці: 10
```

```
Хеш-таблиця:
0: 0: Модуль вектора: 93.03567207474607, Кут: 302.7633629181554, X: 50.34817137042468, Y: -78.2348893912047
1: Null
2: Null
3: 3: Модуль вектора: 17.571173801895192, Кут: 42.03074926231848, X: 13.051615089929031, Y: 11.764416361245669
3: Модуль вектора: 77.75014617891559, Кут: 200.4376749767728, X: -72.85597526645577, Y: -27.149440119758033
3: Модуль вектора: 56.762981783506675, Кут: 235.60433575145, X: -32.06566741214697, Y: -46.83832911588848
3: Модуль вектора: 15.820820362538758, Кут: 325.31833074651155, X: 13.009873976147798, Y: -9.00230726361155
3: Модуль вектора: 18.19162711195431, Кут: 132.78578308023344, X: -12.356830468876778, Y: 13.35080663270312
4: 4: Модуль вектора: 71.75582809163278, Кут: 241.99498238173598, X: -33.69286901542612, Y: -63.35368531210552
5: Null
6: 6: Модуль вектора: 68.39620191697348, Кут: 162.0275989921669, X: -65.05882684715628, Y: 21.10425279272267
7: 7: Модуль вектора: 73.46818230199233, Кут: 300.9585824509209, X: 37.79337874014585, Y: -63.001859767491545
8: Null
9: 9: Модуль вектора: 60.57676930893735, Кут: 165.08117480367412, X: -58.53481989666235, Y: 15.595507031626644
```

```
Введіть значення координати Y для видалення: 0

Видалення елементів з координатою Y менше 0.0:
Видалено елемент на позиції 0
Видалено елемент на позиції 3
Видалено елемент на позиції 3
Видалено елемент на позиції 3
Видалено елемент на позиції 4
Видалено елемент на позиції 7
```

```
Хеш-таблиця після видалення:
0: Null
1: Null
2: Null
3: 3: Модуль вектора: 17.571173801895192, Кут: 42.03074926231848, X: 13.051615089929031, Y: 11.764416361245669
3: Модуль вектора: 18.19162711195431, Кут: 132.78578308023344, X: -12.356830468876778, Y: 13.35080663270312
4: Null
5: Null
6: 6: Модуль вектора: 68.39620191697348, Кут: 162.0275989921669, X: -65.05882684715628, Y: 21.10425279272267
7: Null
8: Null
9: 9: Модуль вектора: 60.57676930893735, Кут: 165.08117480367412, X: -58.53481989666235, Y: 15.595507031626644
```

ВИСНОВОК:

В ході лабораторної роботи №1 було створено програму на мові програмування Java, котра створює хеш-таблицю, вирішує колізії методом роздільного зв’язування, видаляє значення хеш-таблиці за заданими параметрами та виводить її у консоль.