Міністерство освіти і науки України
Львівський національний університет імені Івана Франка
Факультет прикладної математики та інформатики

Кафедра дискретного аналізу та інтелектуальних систем

Лабораторна робота №4

ХЕШ-ТАБЛИЦЯ

з курсу “Алгоритми та структури даних”

Виконав:
студент групи ПМІ-13
Лук’янчук Денис
Євгенійович

Львів – 2024

Хеш-таблиця — це структура даних, що реалізує інтерфейс асоціативного масиву, а саме, вона дозволяє зберігати пари (ключ, значення) і здійснювати три операції: операцію додавання нової пари, операцію пошуку і операцію видалення за ключем.

Складність роботи: O(1)

Алгоритм виконання:

1. Створення класу HashTable:

Визначаємо клас HashTable, який містить приватні поля table (вектор списків пар ключ-значення) і size (розмір таблиці). Визначаємо конструктор HashTable, який ініціалізує розмір таблиці і розмір вектору table.

2. Хеш-функція:

Визначаємо метод hashFunction, який обчислює хеш для ключа, використовуючи просту хеш-функцію (залишок від ділення на розмір таблиці).

3. Перевірка наявності ключа:

Визначаємо метод contains, який перевіряє, чи міститься ключ у вказаній таблиці, проходячи по відповідному списку.

4. Вставка елемента:

Визначаємо метод insert, який перевіряє наявність ключа у таблиці, вставляючи новий елемент у відповідний список, якщо ключ ще не існує.

5. Пошук елемента:

Визначаємо метод search, який здійснює пошук значення за ключем, перебираючи список у відповідному індексі таблиці.

6. Видалення елемента:

Визначаємо метод remove, який видаляє елемент з таблиці, перебираючи список у відповідному індексі.

7. Функція main:

Користувач вводить розмір хеш-таблиці. Користувач вводить послідовно ключі та значення. Програма перевіряє наявність ключа у таблиці перед вставкою. Користувач може виконати пошук значення за введеним ключем. Користувач може видалити елемент за ключем. Програма виводить результати операцій пошуку та видалення елемента.

Приклад

Дано: size = 5, keys = {1, 4, 1, 5, 9}, search key = 1, remove key = 5.

```
Enter size of the hash table: 5
Enter key and value pairs (enter -1 to stop):
Enter key (-1 to stop): 1
Enter value: Apple
Enter key (-1 to stop): 4
Enter value: Pear
Enter key (-1 to stop): 1
Key already exists. Please enter a different key.
Enter key (-1 to stop): 5
Enter value: Banana
Enter key (-1 to stop): 9
Enter value: Peach
Enter key (-1 to stop): -1
Enter key to search: 1
Value corresponding to key 1: Apple
Enter key to remove: 5
After removing key 5, value corresponding to key 5: Key not found
```

Приклад Unit-тесту(*без помилок*)

```
[==========] Running 4 tests from 1 test suite.
[----------] Global test environment set-up.
[----------] 4 tests from HashTableTest
[ RUN      ] HashTableTest.InsertTest
[       OK ] HashTableTest.InsertTest (0 ms)
[ RUN      ] HashTableTest.SearchTest
[       OK ] HashTableTest.SearchTest (0 ms)
[ RUN      ] HashTableTest.RemoveTest
[       OK ] HashTableTest.RemoveTest (0 ms)
[ RUN      ] HashTableTest.ContainsTest
[       OK ] HashTableTest.ContainsTest (0 ms)
[----------] 4 tests from HashTableTest (2 ms total)

[----------] Global test environment tear-down
[==========] 4 tests from 1 test suite ran. (4 ms total)
[  PASSED  ] 4 tests.
```

Приклад Unit-тесту(*з помилкою*)

```
[==========] Running 4 tests from 1 test suite.
[----------] Global test environment set-up.
[----------] 4 tests from HashTableTest
[ RUN      ] HashTableTest.InsertTest
C:\ЫуюЁшЄьш Єр ёЄЁЕъЄеЁш фрэшї\Test\Test.cpp(68): error: Expected equality of these values:
  "Value"
  ht.search(10)
    Which is: "Value1"

[  FAILED  ] HashTableTest.InsertTest (2 ms)
[ RUN      ] HashTableTest.SearchTest
[       OK ] HashTableTest.SearchTest (0 ms)
[ RUN      ] HashTableTest.RemoveTest
[       OK ] HashTableTest.RemoveTest (0 ms)
[ RUN      ] HashTableTest.ContainsTest
[       OK ] HashTableTest.ContainsTest (0 ms)
[----------] 4 tests from HashTableTest (5 ms total)

[----------] Global test environment tear-down
[==========] 4 tests from 1 test suite ran. (7 ms total)
[  PASSED  ] 3 tests.
[  FAILED  ] 1 test, listed below:
[  FAILED  ] HashTableTest.InsertTest

 1 FAILED TEST
```

Висновок: У даному коді була реалізована проста хеш-таблиця. Хеш-таблиця дозволяє зберігати ключ-значення у вигляді пар і швидко виконувати операції вставки, пошуку та видалення. Користувач може ввести розмір хеш-таблиці та послідовно вводити ключі та значення. Програма перевіряє наявність ключа у таблиці перед вставкою нового елемента, щоб уникнути дублювання ключів. Кожен елемент зберігається у відповідному списку залежно від значення хеш-функції для його ключа. Після введення даних користувач може здійснювати операції пошуку та видалення, а результати цих операцій виводяться на екран. Цей приклад демонструє базовий механізм роботи хеш-таблиць і може бути використаний як основа для подальших розширень та оптимізацій.