Міністерство освіти і науки України
ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені ІВАНА ФРАНКА
Факультет прикладної математики та інформатики

Кафедра програмування

ЛАБОРАТОРНА РОБОТА № 4

Хеш-таблиця

з курсу “Алгоритми та структури даних”

Виконав:

Студент групи ПМІ-16

Бевз Маркіян Юрійович

Львів – 2024

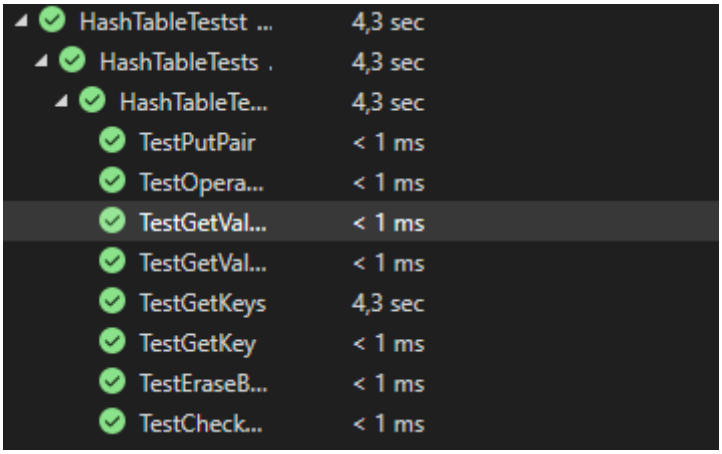
Мета: Ознайомлення з принципом роботи хеш-таблиць та реалізація класу, який надає основні операції з даними: додавання, видалення, пошук та оновлення.

Принцип роботи хеш-таблиці:

1. **Додавання елементу (Insert):** Новий запис додається до хеш-таблиці.
2. **Видалення елементу (Delete):** Існуючий запис видаляється з хеш-таблиці.
3. **Пошук елементу (Search):** Пошук запису за ключем або іншими параметрами.
4. **Оновлення елементу (Update):** Зміна значень полів в існуючому записі.
5. **Отримання розміру таблиці (GetSize):** Повертається кількість записів в хеш-таблиці.
6. **Перевірка наявності елементу (IsEmpty):** Перевірка, чи є хеш-таблиця порожньою.
7. **Отримання значення за ключем (GetValueByKey):** Повертається значення поля за вказаним ключем.
8. **Отримання ключа за значенням (GetKeyByValue):** Пошук ключа за значенням певного поля.
9. **Отримання всіх ключів (GetAllKeys):** Повертається масив усіх ключів у хеш-таблиці.
10. **Отримання всіх значень (GetAllValues):** Повертається масив усіх значень у хеш-таблиці.
11. **Хеш-функція (HashFunction):** Цей метод приймає ключ і обчислює хеш-значення для нього.
12. **Зміна розміру таблиці (Resize):** Цей метод використовується для зміни розміру хеш-таблиці.

Хід роботи: Після детального вивчення принципу роботи хеш-таблиць та їх основних операцій, реалізував клас **HashTable** для обробки та збереження даних. В цьому класі були написані методи для додавання, видалення, пошуку, оновлення та отримання розміру таблиці та інших операцій. Нижче наведено результат виконання програми, тести та код самих тестів.

```
271236 ---> Bevz Markiyan
276564 ---> Mraka Olga
274345 ---> Shulhak Danylo
274346 ---> Proz Mykhailo-Ihor
276543 ---> Kovalchuk Julia
272127 ---> Mekheda Ksenia
276544 ---> Prozay Ivan
271422 ---> Lopatinskiy Oleksa
Who you want to terminate?
271422
Lopatinskiy Oleksa's key is 271422
271236 ---> Bevz Markiyan
276564 ---> Mraka Olga
274345 ---> Shulhak Danylo
274346 ---> Proz Mykhailo-Ihor
276543 ---> Kovalchuk Julia
272127 ---> Mekheda Ksenia
276544 ---> Prozay Ivan
Empty table declaration:
There is nothing to delete, empty tableError: Key 121 not found in the table.
```




```cpp
TEST_METHOD(TestPutPair)
{
    // Arrange
    HashTable table;

    // Act
    table.PutPair(1, "Value1");
    table.PutPair(2, "Value2");

    // Assert
    Assert::AreEqual(std::string("Value1"), table.GetValue(1));
    Assert::AreEqual(std::string("Value2"), table.GetValue(2));
}
```

```
TEST_METHOD(TestEraseByKey)
{
    // Arrange
    HashTable table;
    table.PutPair(1, "Value1");
    table.PutPair(2, "Value2");

    // Act
    table.EraseByKey(1);

    // Assert
    Assert::IsFalse(table.CheckByKey(1));
    Assert::AreEqual(std::string("Value2"), table.GetValue(2));
}

TEST_METHOD(TestCheckByKey)
{
    // Arrange
    HashTable table;
    table.PutPair(1, "Value1");

    // Assert
    Assert::IsTrue(table.CheckByKey(1));
    Assert::IsFalse(table.CheckByKey(2));
}

TEST_METHOD(TestGetKey)
{
    // Arrange
    HashTable table;
    table.PutPair(1, "Value1");

    // Act
    int key = table.GetKey("Value1");

    // Assert
    Assert::AreEqual(1, key);
}
```

```
TEST_METHOD(TestGetValue)
{
    // Arrange
    HashTable table;
    table.PutPair(1, "Value1");

    // Assert
    Assert::AreEqual(std::string("Value1"), table.GetValue(1));
}
```

```
TEST_METHOD(TestGetValues)
{
    HashTable table;
    table.PutPair(1, "Value1");
    table.PutPair(2, "Value2");

    // Act
    int size = 2; // Встановлюємо розмір масиву
    std::string* values = table.GetValues(size);

    // Assert
    Assert::AreEqual(2, size);
    Assert::AreEqual(std::string("Value1"), values[0]);
    Assert::AreEqual(std::string("Value2"), values[1]);

    // Звільнення пам'яті
    delete[] values;
}

TEST_METHOD(TestOperatorBracket)
{
    // Arrange
    HashTable table;
    table.PutPair(1, "Value1");

    // Act
    std::string& value = table[1];

    // Assert
    Assert::AreEqual(std::string("Value1"), value);
}
```

Висновки: Я ознайомився з принципом роботи хеш-таблиць та реалізував клас хеш-таблиці, який надає можливість виконувати основні операції з даними: додавання, видалення, пошук та їх оновлення.