**Київський національний університет імені Тараса Шевченка**

**Факультет комп’ютерних наук та кібернетики**

**Алгоритми та складність**

**Завдання № 1**

**Звіт**

**Виконав:**

студент групи К-29

Грищенко Юрій Анатолійович

**Київ-2020**

**Умова задачі**

Реалізуйте ідеальне хешування.

Предметна область: відділ кадрів (варіант 3). Обєкти: відділення фірми, працівники. Маємо множину відділень, у кожному відділенні зберігається множина працівників.

**Опис алгоритмів**

**Хеш-таблиця** — 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.

Виконання операцій в хеш-таблиці починається з обчислення хеш-функції від ключа. Отримане хеш-значення *i* = hash(*key*) відіграє роль індексу в масиві *H*. Після цього операція (додавання, видалення, пошук) перенаправляється об'єктові, який зберігається у відповідній комірці масиву *H*[*i*].

Важлива властивість хеш-таблиць полягає в тому, що, при деяких розумних припущеннях, всі три операції (пошук, вставлення і видалення елементів) зазвичай виконується за час *O*(1).

Ситуація, коли для різних ключів отримується одне й те саме хеш-значення, називається колізією. Такі події непоодинокі — наприклад, при додаванні в хеш-таблицю розміром 365 комірок усього лише 23-х елементів ймовірність колізії вже перевищує 50 відсотків (якщо кожний елемент може з однаковою ймовірністю потрапити в будь-яку комірку) — так званий “парадокс днів народження”. Через це механізм розв'язання колізій — важлива складова будь-якої хеш-таблиці.

**Ідеальне хешування** дозволяє повністю уникнути колізій, проте має специфічні властивості:

* Множина ключів статична – не змінюється після збереження в таблицю.
* Ο(1) звертань до пам’яті в найгіршому випадку.
* Перший рівень хешування: n ключів хешуються в m комірок за допомогою універсальної хеш-функції h.
* Другий рівень хешування: для кожної комірки своя вторинна хеш-таблиця зі своєю універсальною хеш-функцією, вибраною так, щоб уникнути колізій; її розмір – квадрат кількості ключів, захешованих в комірку.
* Очікувана загальна пам’ять під таку структуру O(n).

В нашому випадку уніваерсальна хеш-функція матиме вигляд:

ha,b(k)=((ak+b) mod p) mod m.

де p — просте, а є{1,2,...,p–1}, b є{0,1,...,p–1}



**Модулі програми:**

У своїй програмі я розбив модулі на дві категорії: ті, що стосуються саме предметної області (HumanResources.\*) і ті, що стосуються ідеального хешування (PerfectHashSet.\*)

PerfectHashSet<T> реалізовано як template-клас, тобто він може зберігати обєкти будь-якого класу. Єдина умова: для класу Т має бути реалізована функція int32\_t getHashValue(T object).

У нашому випадку необхідно зберігати інформацію про працівників фірми, відповідне хеш-значення залежить від імені працівника. Реалізація getHashValue схожа на функцію, яка раніше використовувалась для алгоритма Рабіна-Карпа.

Основні операції над PerfectHashSet<T>: створення таблиці, перевірка на наявність елемента, пошук елемента за відомим ключем.

**Інтерфейс користувача, тестові приклади:**

Програма не має інтерактивного режиму, множина значень у хеш-таблиці задана у функції main. Програма виводить інформацію щодо структури хеш-таблиці (схоже на ілюстрацію, подану раніше), а також показує результати деяких операцій.

Таблиця зберігає працівників з іменами Pasha, Vasilii, Pavel, John, Markus, David, Petro. Ми бачимо, що у первинній таблиці зберігається 7 вторинних таблиць, для кожної з них бачимо підібрані коефіцієнти відповідної хеш-функції.

a = 514229, b = 55, m1 = 6972593, m2 = 7

Secondary hash set 0: null

Secondary hash set 1: null

Secondary hash set 2:

a = 920115, b = 5285506, m1 = 6972593, m2 = 1

0: { Name: Pasha } (Hash value: 5312377)

Secondary hash set 3:

a = 3208286, b = 3726740, m1 = 6972593, m2 = 1

0: { Name: Vasilii } (Hash value: 1915222)

Secondary hash set 4:

a = 1531633, b = 329080, m1 = 6972593, m2 = 1

0: { Name: Pavel } (Hash value: 5508228)

Secondary hash set 5:

a = 4748700, b = 4751720, m1 = 6972593, m2 = 4

0: { Name: John } (Hash value: 721067)

1: { Name: Markus } (Hash value: 1315058)

2: { Name: } (Hash value: 0)

3: { Name: } (Hash value: 0)

Secondary hash set 6:

a = 241835, b = 373968, m1 = 6972593, m2 = 4

0: { Name: David } (Hash value: 336555)

1: { Name: } (Hash value: 0)

2: { Name: Petro } (Hash value: 2763421)

3: { Name: } (Hash value: 0)

Hash set contains Yurii? false

Hash set contains Markus? true

Hash set contains Vasilii? true

Element with value 1915222 = { Name: Vasilii }

**Список використаних джерел:**

* https://uk.wikipedia.org/wiki/Геш-таблиця
* Лекція 1 з курсу “Алгоритми та складність” Шкільняк О.С.