ADS Homework 11

Name - Tewodros Adane
Email - tadane@jacobs-university.de

Problem 11.1 – Hash Tables

a) Insert the sequence <3, 10, 2, 4> into a hash table with 5 slots that uses double-hashing for open addressing.

$$h_1(k) = k \bmod 5, \quad \text{and} \quad h_2(k) = 7k \bmod 8$$

To find the positions we will use the formula:

$$pos(k) = (h_1(k) + i * h_2(k)) \bmod 5 \quad ,$$

$$initially\ i = 0\ and\ increments\ by\ 1\ everytime\ there\ is\ a\ collision.$$

The following are the steps the algorithm will perform:

`Inserting 3:`

$$h_1(3) = 3 \bmod 5 = 3, \quad and \quad h_2(3) = (7 * 3) \bmod 8 = 5$$

Pos(3) = (3 + 0 * 5) mod 5 = 3, No collision at 3.

0	\
1	\
2	\
3	3
4	\

`Inserting 10:`

$$h_1(10) = 10 \bmod 5 = 0, \quad and \quad h_2(10) = (7 * 10) \bmod 8 = 6$$

Pos(10) = (0 + 0 * 6) mod 5 = 0, No collision at 0.

0	10
1	\
2	\
3	3
4	\

`Inserting 2:`

$h_1(2) = 2 \bmod 5 = 2, \quad and \quad h_2(2) = (7 * 2) \bmod 8 = 6$

Pos(2) = (2 + 0 * 6) mod 5 = 2, No collision at 2.

0	10
1	\
2	2
3	3
4	\

`Inserting 4:`

$h_1(4) = 4 \bmod 5 = 4, \quad and \quad h_2(4) = (7 * 4) \bmod 8 = 4$

Pos(4) = (4 + 0 * 4) mod 5 = 4, No collision at 4.

0	10
1	\
2	2
3	3
4	4

b) The template implementation of a Hash Table is in the file "*HashTable.h*", and the usage is in "main.*cpp*".

- This implementation uses the Multiplication method to calculate the hash values of a given key. Multiplication method was chosen because it does not depend on the number of slots(**m**), this is important because it allows the hash table to have variable size and still use the same hash function for s**m**.

Problem 11.2 – Greedy Algorithms

a) Proof by contradiction:

Let S be a set of activities to choose form.

$S = \{a_1(1,5), a_2(4,7), a_3(6,10)\}$

If we chose activities based on the shortest duration, we would first choose a_2 and then stop because there are no other activities that won't overlap with a_2. The solution would be:

$$\{a_2(4,7)\}$$

However, there is a better solution with two activities:

$$\{a_1(1,5), a_3(6,10)\}$$

Therefore, selecting the activity with the shortest duration does not always return the globally optimal solution.

□

b) The implementation and the usage of a Greedy Activity Selection algorithm that selects activities based on the latest starting time is in the file "ActivitySelectionGreedy.*cpp*".