BLG381E, Advanced Data Structures, Fall 2012

Project 3

Handed out: 06.12.2012

Due: 21.12.2012, until 5.00 PM

Linear Probing vs Double Hashing

a) **(20 pts)** Write a C++ function to implement *linear probing strategy* for key insertion to a hash table. Linear probing hash function is given as follows,

$$h(k,i) = (h'(k) + i)\ mod\ m \text{ where } h'(k) = k \text{ and } i \in [0, m-1]$$

Your function must take two parameters: address of the hash table (can be represented as an array of size m) and key to be inserted (k). In the function, linear probing strategy will be performed until finding an empty slot for the given key in the hash table. All the collisions must be printed out on the screen with the values of k and $h(k,i)$.

b) **(30 pts)** Write a C++ function to implement *double hashing strategy* for key insertion to a hash table. The corresponding hash function is given as follows,

$$h(k,i) = \big(h_1(k) + i.h_2(k)\big)\ mod\ m \text{ where } h_1(k) = k,\ h_2(k) = R - k\ mod\ R \text{ where } R \text{ is the largest prime number smaller than } m \text{ and } i \in [0, m-1]$$

Your function must take three parameters: address of the hash table (can be represented as an array of size m), key to be inserted (k) and value of R. In the function, double hashing strategy will be performed until finding an empty slot for the given key in the hash table. All the collisions must be printed out on the screen with the values of k, $h_1(k)$, $h_2(k)$ and $h(k,i)$.

c) **(15 pts)** Consider that the following keys are given to be inserted into a hash table of size 13.

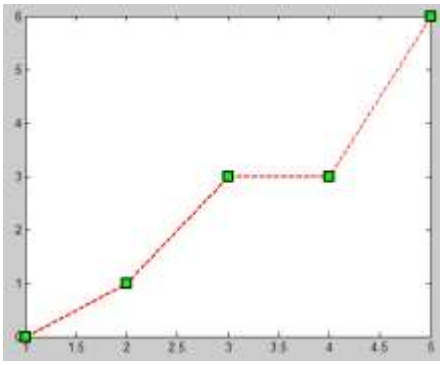
$$key\_set = \{1, 27, 5, 42, 18, 9, 35, 72, 49, 12, 92\}$$

Run your program to insert the given set of keys in the given order. Show the resulting hash table after performing linear probing and double hashing strategies and compare the results. Fill in the table below for indicating number of collisions for each key value using different probing strategies.

	# collisions										
Probing Strategy	**1**	**27**	**5**	**42**	**18**	**9**	**35**	**72**	**49**	**12**	**92**
linear probing											
double hashing											

d) **(35 pts)** Use the integers given in the files *'x1.txt'* and *'x2.txt'* for testing different probing strategies that you implemented in **a** and **b**. For each dataset, compare the number of collisions for each probing strategy and for every element inserted to the hash table of size of 1327. Plot the sum of all collisions occurred so far when inserting an element and evaluate the resulting plot.

e.g., If you had {0,1,2,0,3} collisions for the first five elements, you should plot x={1,2,3,4,5} and y={0,1,3,3,6}.



e) (BONUS: 30 pts) The following H is a family of universal hash functions for a hash table of size m.

$h_{a,b} = \big((ak + b) mod\ p\big)\ mod\ m$ where p is a prime number bigger than any key value to be inserted

$H = \{h_{a,b} | 1 \leq a \leq p-1, 0 \leq b \leq p-1\}$

Implement a C++ function to select a hash function $h_{a,b}$ from H randomly (p must be chosen w.r.t. the key values in the key_set) and return its parameters and another C++ function for using that hash function as a probing strategy. Using that insertion function, insert the given set of keys in **c** to a hash table of size 13 and show all collisions and the resulting hash table.

Detailed Instructions:

- All your code must be written in C++ and able to compile and run on linux using g++.
- Submissions will be done through the Ninova server. You must submit all your programs and header files. You must also submit a softcopy report.
- For universal hashing, you can refer to the course book, you can also check for other sources in the internet. For those who are further interested, there is also a video lecture on the topic: http://videolectures.net/mit6046jf05_leiserson_lec08/