Problem 9.1

a) Sequence <3,10,2,4>, m=5, $h_1(k)$= k mod 5, $h_2(k)$= 7k mod 8
Double hashing uses the hash function h(k,i) = ($h_1(k)$ + i*$h_2(k)$) mod 5

Insert 3 >> 3 mod 5 = 3, i=3.
So, 3 will be inserted at positon 3.

0	1	2	3	4
			3	

Insert 10 >> 10 mod 5 = 0, i=0.
So, 3 will be inserted at positon 0.

0	1	2	3	4
10			3	

Insert 2 >> 2 mod 5 = 2, i= 2
So, 2 will be inserted at positon 2.

0	1	2	3	4
10		2	3	

Insert 4 >> 4 mod 5 = 4, i=4
So, 3 will be inserted at positon 4.

0	1	2	3	4
10		2	3	4

No collision occurred for inserting this sequence with h_1 function.

b) The implementation for this problem is in Hash.cpp.

Problem 9.2

a) For instance, there are 5 activities with start time and finish time as shown in the table below. We have to choose one activity with the shortest duration.

Activity	a1	a2	a3	a4	a5
Start	0	2	5	6	3
Finish	2	7	8	13	4

The greedy algorithm would return the first activity (0,2) as the start and end timings are earlier than other activities and the duration is short compared with a2 and a3, and it

could only return one activity. The greedy algorithm will not go and check until a5, which has shorter duration (4-3=1) than a1(2-0=2). Hence, the greedy algorithm will fail to produce the globally optimal solution.

b) The implementation for this problem is in Greedy.cpp.

References

C Program for Activity Selection Problem | Greedy Algo-1. (2018, December 11). Retrieved April 26, 2019, from https://www.geeksforgeeks.org/activity-selection-problem-greedy-algo-1/

(I used this website to get some hints for Problem 9.2b.)

Chhavi. (2018, October 26). Implementing own Hash Table with Open Addressing Linear Probing in C. Retrieved April 26, 2019, from https://www.geeksforgeeks.org/implementing-hash-table-open-addressing-linear-probing-cpp/

(I used this website to get some hints for Problem 9.1b.)

Cplusplus. (n.d.). Std::sort. Retrieved April 26, 2019, from http://www.cplusplus.com/reference/algorithm/sort/?kw=sort

(I used this website to make sure that sort function in C++ library's time complexity is O (n log n) in all cases.)