Name: Brian Sherif

Homework 9

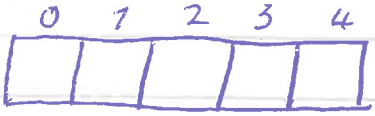
Problem 1

a) Double hashing works as follows.

while no collisions occur the main hash function is used. here denoted as $h_1(k)$ where k is the key. when a collision occurs we compute the position using 2 hash functions this way $(h_1(k) + i * h_2(k)) \%$ size of array where i is incremented with every collision.

In the following problem we have an array of size 5 where we must insert $< 3, 10, 2, 4 >$

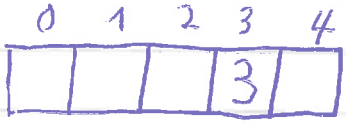
this is the initialized empty array



hash function $\begin{cases} h_1(k) = K \bmod 5 \\ h_2(k) = (7 \cdot K) \bmod 8 \end{cases}$

Insert 3:

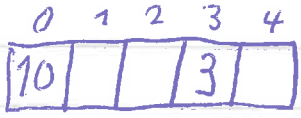
$3 \bmod 5 = 3$



No collision. 3 inserted in index 3

Insert 10:

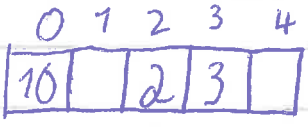
$10 \bmod 5 = 0$



No collision. 10 inserted in index 0

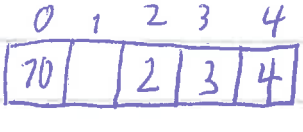
Insert 2:

$2 \bmod 5 = 2$



No collision.

Insert 4: $4 \bmod 5 = 4$



No collision,

No collisions occurred

Problem 2

b) The following function is a Selection function which checkes for overlays and is recursively called to go thru the list.

selection (list 1, selected)

// following loop is to place find (largest start time
// at the end of the array.

for (i = length of list 1; i--) {
Largest Starting time = list 1[i];
if (largest starting time < list 1[i-1]. start time) {
Largest Starting time = list 1[i-1];
}}

// Set last Element in array as Biggest Starting time

last element of arr = Biggest Starting time;

head = first element of list 1;

tail = all other elements;

// check for collisions with selected
if collision with Between Elements in selected and list 1
exist delete from list 1 // if (true) {return selection (tail; selected)

return (1 + selection (tail · selected) + head] selection (tail;
or 1 + selection (tail, selected) + head [selection (tail; selected) selected)
// return the greater list of both ; }

Problem 2

b.) Assume that all activities are present in a list (in an unsorted manner) (list 1).

Create a new empty list for selected values. (selected)

- This function is used to check for overlay in time.

```
bool overlay (selected, other) {
if (selected.start ≤ other.start and
    selected.end ≥ other.start) {
  return true; }
if (selected.start < other.end and selected.end ≥
                                    other.end) {
  return true; }
else { return false }}
```

- This function will be used to ensure no conflicts between selected and list 1 occurs.