CS245 Database System Principles - Winter 2001

- PLEASE write your serial number on the top of your first page. If you have not received your serial number by e-mail, send a message to **orkut@stanford.edu**.
- This assignment is due in class on Thursday, Feb 8th.
- State all assumptions.
- Subscribe to **cs245-win01** to receive clarifications and changes.
- Email questions to **cs245ta-win01@lists.stanford.edu**.

Assignment 4

Problem 1 (30 points)

Consider a *linear* hash structure where buckets can hold up to three records. Initially the structure is empty. Assume that the threshold value is 2. (i.e., when the average number of keys per non-overflow bucket is greater than 2, we allocate another bucket). The hashed key values are as follows (in binary):

```
h(Ashley)= [00010]
h(Brian)=  [00011]
h(Chris)=  [00101]
h(Daniel)= [00111]
h(Ethel)=  [01011]
h(Frank)=  [10001]
h(George)= [10011]
h(Harold)= [10111]
h(Jeff)=   [11101]
h(Karen)=  [11111]
```

We insert records in the following order:
Ashley, Karen, Brian, Jeff, Chris, Harold, Daniel, George, Frank, Ethel

a) Show the structure after all these records have been inserted. Use a diagram like the one used in class.

b) Suppose now, we decided to use the opposite bits (high order bits) of the hash but *do not change anything else in the algorithm*. Show the structure after the records are inserted in the same order as part a.

c) What is the problem that occurred (if any) because we used the high order bits instead of the low order ones? Is this problem serious?

Problem 2 (50 points)

Consider an *extensible* hash structure where buckets can hold up to three records. Initially the structure is empty. The hashed key values are same as in Problem 1.

a) We insert records in the following order:
Ashley, Karen, Brian, Jeff, Chris, Harold, Ethel, George, Frank, Daniel
Show the structure after all these records have been inserted.

b) Suppose, instead we insert records in the following order:
Ashley, Brian, Chris, Daniel, Ethel, Frank, George, Harold, Jeff, Karen
Show the structure after all these records have been inserted.

c) Does the final structure depend on the order in which records are inserted? Explain.

d) Suppose now, we decided to use the opposite bits (low order bits) of the hash but *do not change anything else in the algorithm*. Show the structure after the records are inserted in the same order as *Problem 2 - Part a*.

e) What is the problem that occurred (if any) because we used the low order bits instead of the high order ones? Is this problem serious?

Problem 3 (20 points)

Consider an extensible hash table where 1000 buckets are actually allocated at this point.

a) What is the size (in the number of entries) of the smallest possible directory in this case?

b) What is the size (in the number of entries) of the largest possible directory in this case?