Name 1:
Name 2:

Suppose that each heap block can hold 10 tuples, and each index block can hold 100 index records (for B+tree indexes). Nodes of the index are 70% full. The relation contains 1 million records.

Determine, for each of the structures determine:

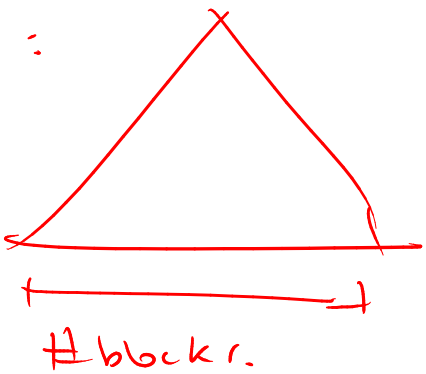
i) The total number of blocks needed by the heap and the leaves of the index.
ii) The average number of disk I/Os needed to retrieve a given search key.

Assume that nothing is in memory initially, and that the search key is the primary key of the table.

a) The table is an unsorted heap, packed 10 to a block. The B+tree is dense.
b) The table is a sorted heap with 10 records per block. The B+tree is sparse and only the first record in each block is in the index.

a) Table : #blocks = $\frac{\#\text{records}}{\#\text{records per page}} = \frac{10^6}{10} = 10^5$

Index :



Index is dense so one index record per tuple.

$\Rightarrow \frac{10^6}{70}$ blocks in leaves of index

the height of the tree is $\lceil \log_{70}(10^6) \rceil$

$$h = \left\lceil \frac{\log_{10}(10^6)}{\log_{10}(70)} \right\rceil = \left\lceil \frac{6}{1.8} \right\rceil = 4.$$

How expensive:

- Using index, $h+1$.
- Using seq scan

  10^5 blocks if not found

  $\frac{10^5}{2}$ block, average if found.

  (because it is a primary key)

  10^5 worst case

b) Sparse Index.

Index will only contain one record per block.

We need $\frac{10^5}{70}$ blocks in the index. Heap is 10^5 also.

$$h = \left\lceil \frac{10^5}{70} \right\rceil = \left\lceil \frac{5}{1.8} \right\rceil = 3$$

So we need to read $h+1 = 4$ blocks

whether the heap contains the value or not