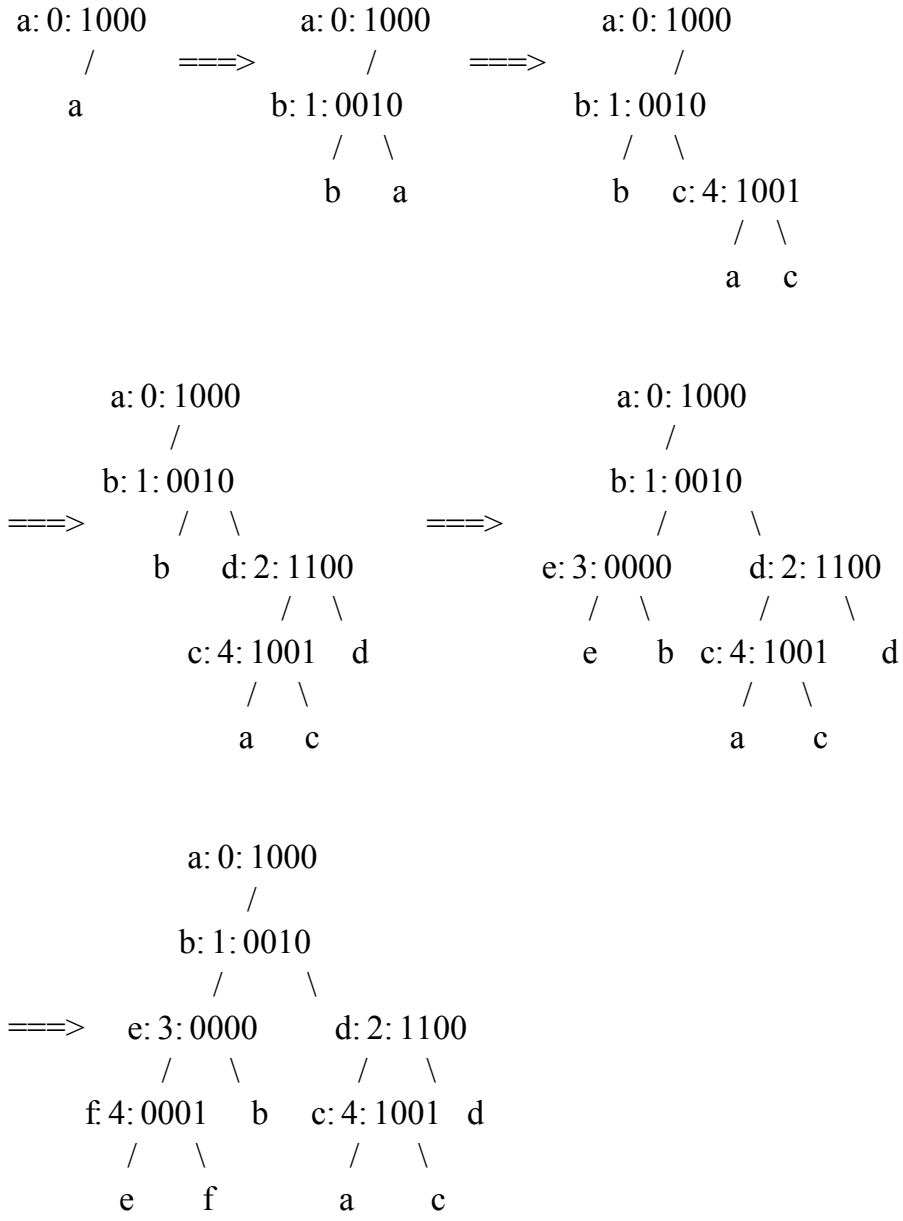
Sample Solution - Sample 2, Exam 3

1.

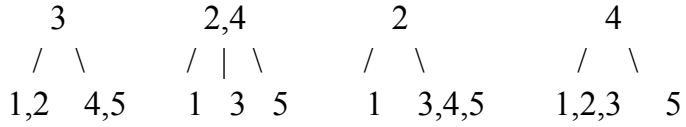


2.

(a) Searching for the insertion position takes h accesses (when the key is not there). We need h splits as we go back up the tree (cascading split in worst case). Each split takes 2 disk accesses (to write the new nodes). When we get back to the root and split it, there are 3 accesses: 2 for splitting the node (already accounted for) and 1 for writing the new root.
Therefore, the total number of disk accesses is $h + 2h + 1 = 3h + 1$.

(b) B-trees of order 3:

B-trees of order 4:



3.

```
insert(key) {
  key = concatenate(key, ' ');
  **p = &root;
  int i = 1;
  for( ; p is a branch node; p = &((*p)->link[c]). i++) {
    c = ith character of key;
  }
  if (*p == NULL) *p = new node(key);
  else {
    if (p->key == key) return;
    else {
      *temp = *p;
      *p = NULL;
       for ( ; ; p = &((*p)->link[c], i++) {
         *p = new branch_node( );
         c = ith character of key;
         c2 = ith character of temp->key;
         if (c != c2) {
           (*p)->link[c] = new node(key);
           (*p)->link[c2] = new node(temp->key);
           return;
} } } } }
```

4.

(a) Suffix Tree:

The problem of determining if string P is a substring of S is handled by: build a suffix tree from S and then search for P in the suffix tree (upto depth | P |).

(b) Bloom Filter:

A database master file is too large to fit in memory, so a differential file

is often used. A bloom filter avoids search time between the files by attempting to answer the question: "is the key present in the differential file?". The bloom filter will give 1 of 2 answers to the posed question: No or maybe.

(c) Quad Tree:
For image processing we need to look at points in space. A quad tree helps build relevant boundaries around points(cells) allowing for fast processing.

5. Followings are used for parts (a) and (b):
let m = memory in the system
let x = memory size
let y = address of memory
since we caanot repeat values, we use a transformation function,
x = (size of memory request) * m + memory address.

(a) Describe a best-fit scheme:
min x in rectangle (L, R, top) min to the left

(b) Describe a first-fit scheme:
min y in rectangle range(x0, x1) bottom most point