CS 321 Data Structures (Fall 2016)

Homework #2 (92 points), Due date 11/03/2016 (Thursday)

- **Q1(20 points): Basic Data Structures: Stacks, Queues, Linked Lists and Trees**

  (a)(7 points) Please write a pseudocode for `List-Move(x, y)`. This procedure is to move an existing nodes x and insert it right before another existing node y in a doubly non-circular list without a dummy head.

```
List-Move(x, y)
{
```

How to fix the pointers

```
x.prev.next = x.next
x.next.prev = x.prev
y.prev.next = x
x.prev = y.prev
x.next = y
y.prev = x
```

```
}
```

  (b)(6 points) Given a binary tree T and a node x in the tree, please write a recursive method `Depth(x)` that returns the depth of the node x in the tree.

```
Depth(x)
{
```

```
if (x == null) return -1;
else return 1 + depth(x.parent);
```

```
}
```

(c)(7 points) Please write a non-recursive procedure to print all nodes for a binary tree in a level order sequence.

```
Level-Order-Print(r)  // r is the root of the binary tree
{
```

The solution is on the slides

(Breadth-First Traversal)

```
}
```

7+(1+6)
7+(1+2) =2
7+(1+4) = 12%8 =4
0+(1+4)

- **Q2(20 points): Hashing**
  Suppose we would like to insert a sequence of numbers into a hash table with table size 8 using the three open addressing methods, with the primary hash function $h_1(k) = k \bmod 8$, the secondary hash function $h_2(k) = 1 + (k \bmod 7)$, and the constants $c_1 = c_2 = 1/2$ (in quadratic probing).

  7 7 7 7 0

  (a)(10 points) If the sequence of numbers is $< 15, 55, 23, 39, 32 >$, please successively insert these numbers into the following tables.

index	linear	quadratic	double
0	55	55	32
1	23	32	
2	39	23	23
3	32		
4			39
5		39	
6			55
7	15	15	15

1/2 i + 1/2 i²

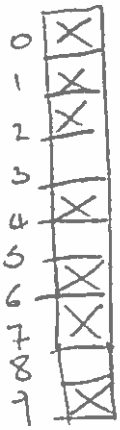
i	#put
i=1	1
i=2	3
3	1.5+1.5=6

7+6

  (b)(3 points) For the hashing functions and table size we used in part (a), does the linear probing fully utilize the table? How about the quadratic probing and double hashing ?

  Linear = Y
  Quadratic = Y ($c_1 = c_2 = 1/2$ ∧ $m = 2^3 = 2$)
  Double = N (m is not prime)

  (c)(7 points) A hash table with size 10 stores 7 elements. These 7 elements are stored in T[0], T[1], T[2], T[4], T[6], T[7], T[9]. Suppose that all the other entries contain no "deleted" flag. An entry has a "deleted" flag means that this entry stored an element before, but the element has already been deleted. If we would like to search an element with a key k and assume the linear probing technique is used, what is the expected number of probes for an unsuccessful search?



$\alpha = \frac{7}{10}$

0	4	7
1	3	9
2	2	10
3	1	12
4	2	13
5	1	15
6	3	16
7	2	18
8	1	
9	5	

Expected $\frac{1}{1-\alpha} = \frac{1}{0.3} = 3.33$

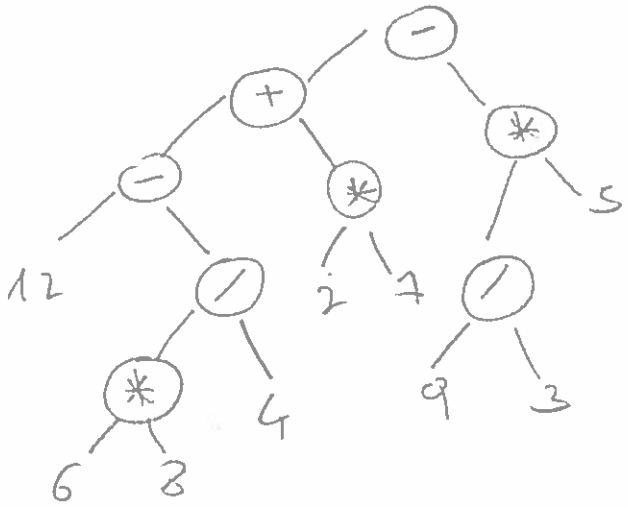
$\frac{24}{10} = 2.4$

- **Q3(12 points): Expressions and Expression Trees**

For a given in-fix expression as below:

$$12 - [6 * 8/4] + (2 * 7) - (9/3) * 5)$$

(a)(6 points) What is the corresponding expression tree?



(b)(6 points) What are the corresponding pre-fix and post-fix expressions?

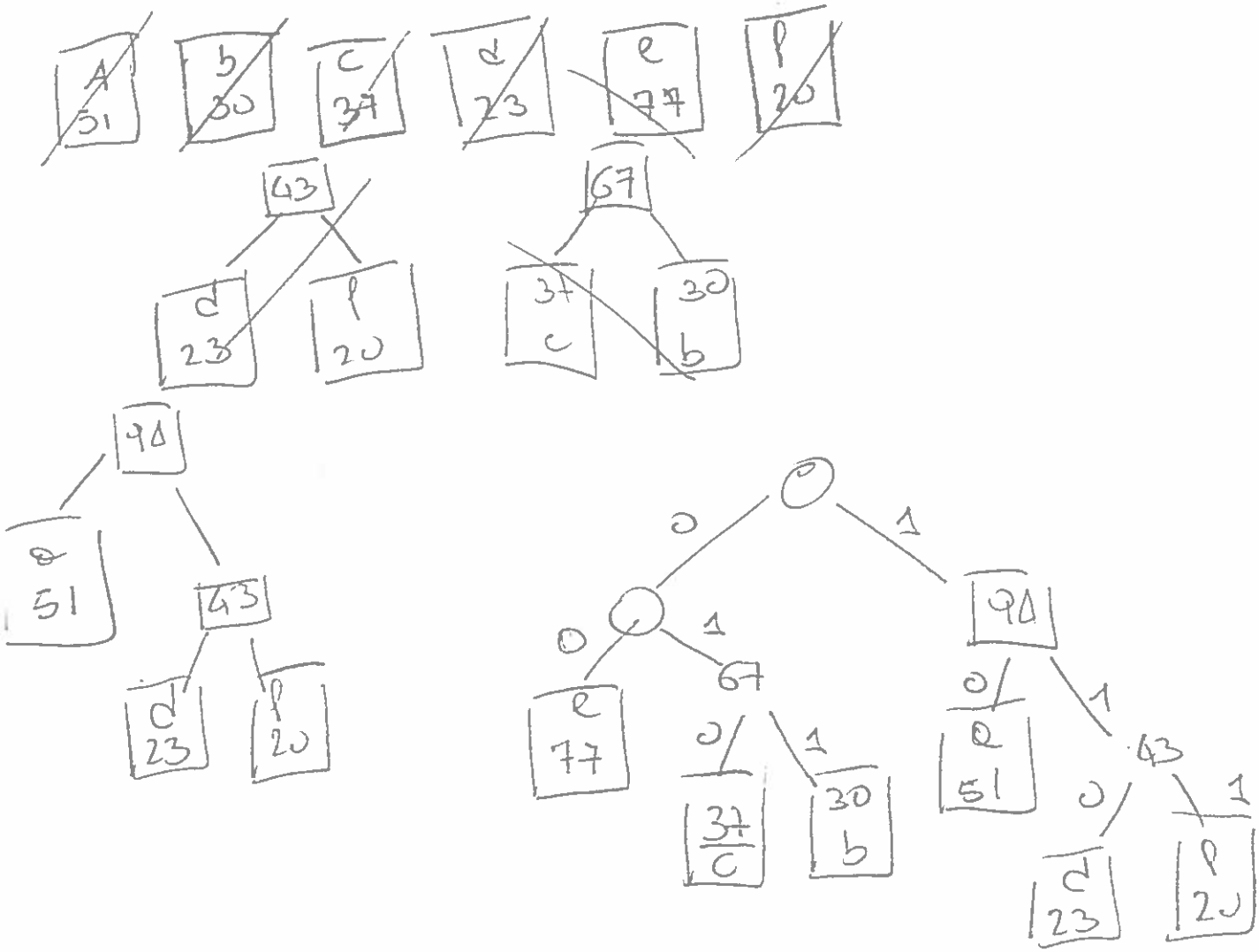
pre-fix

- + - 12 / * 6 8 4 * 2 7 * / 9 3 5

post-fix

12 6 8 * 4 / - 2 7 * + 9 3 / 5 * -

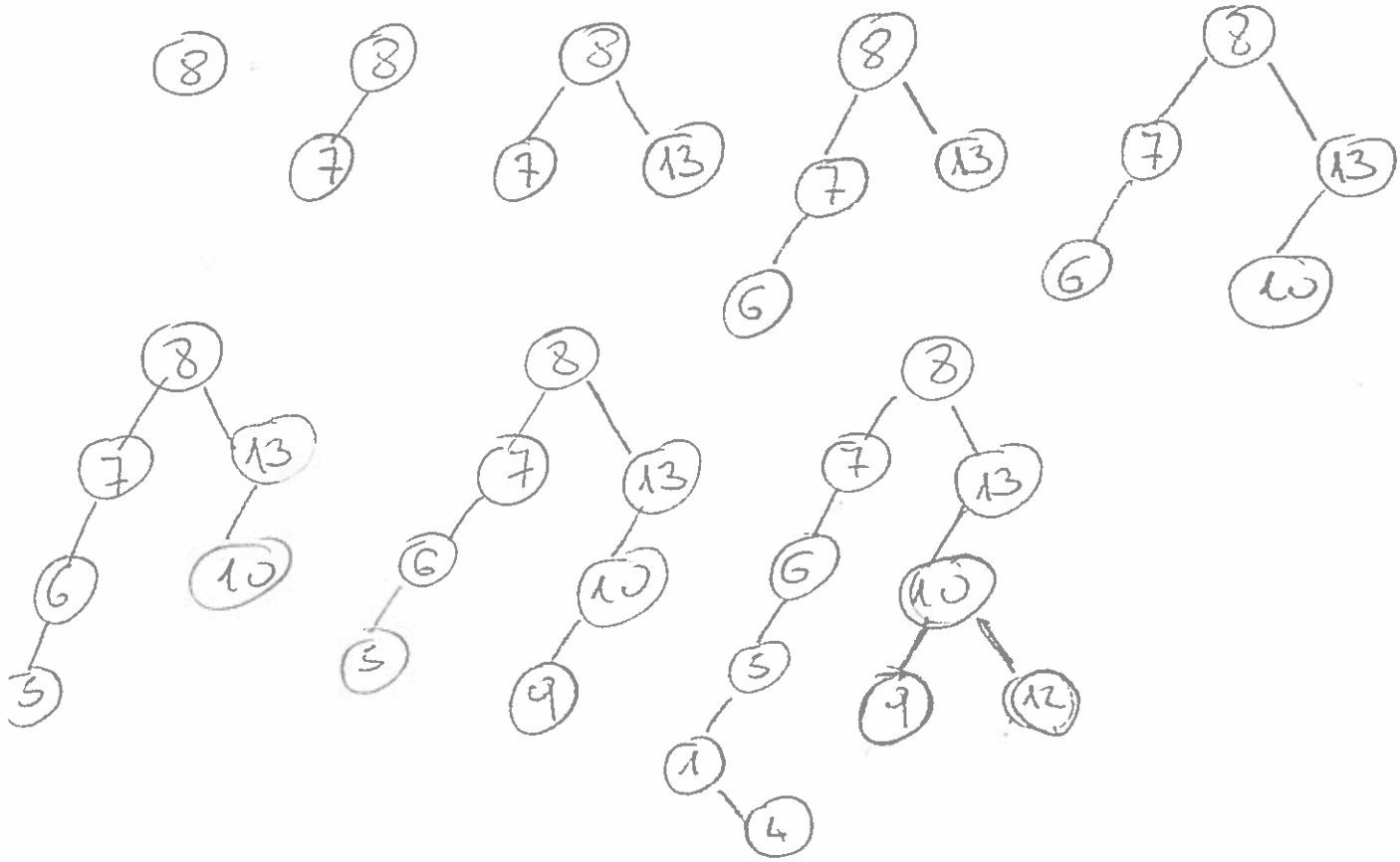
- **Q4(10 points): Huffman Trees** Given a text file with only six different characters $\{a, b, c, d, e, f\}$, the frequencies of these characters in the file are $\{(a : 51), (b : 30), (c : 37), (d : 23), (e : 77), (f : 20)\}$. Based on these frequencies, please construct a Huffman tree with only six leaf nodes (one for each of these six characters). Please show the intermediate forests on each step.



- **Q5(10 points): Binary Search Trees**

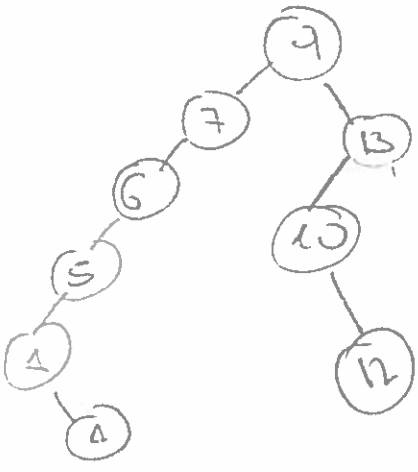
  For a given input array $A : < 8, 7, 13, 6, 10, 5, 9, 1, 12, 4 >$,

  (a)(6 points) What is the resulting binary search tree after inserting the numbers in the list to an initially empty tree?



  (b)(4 points) From the tree you have built in part (a), what is the resulting tree after deleting the value 8 ?

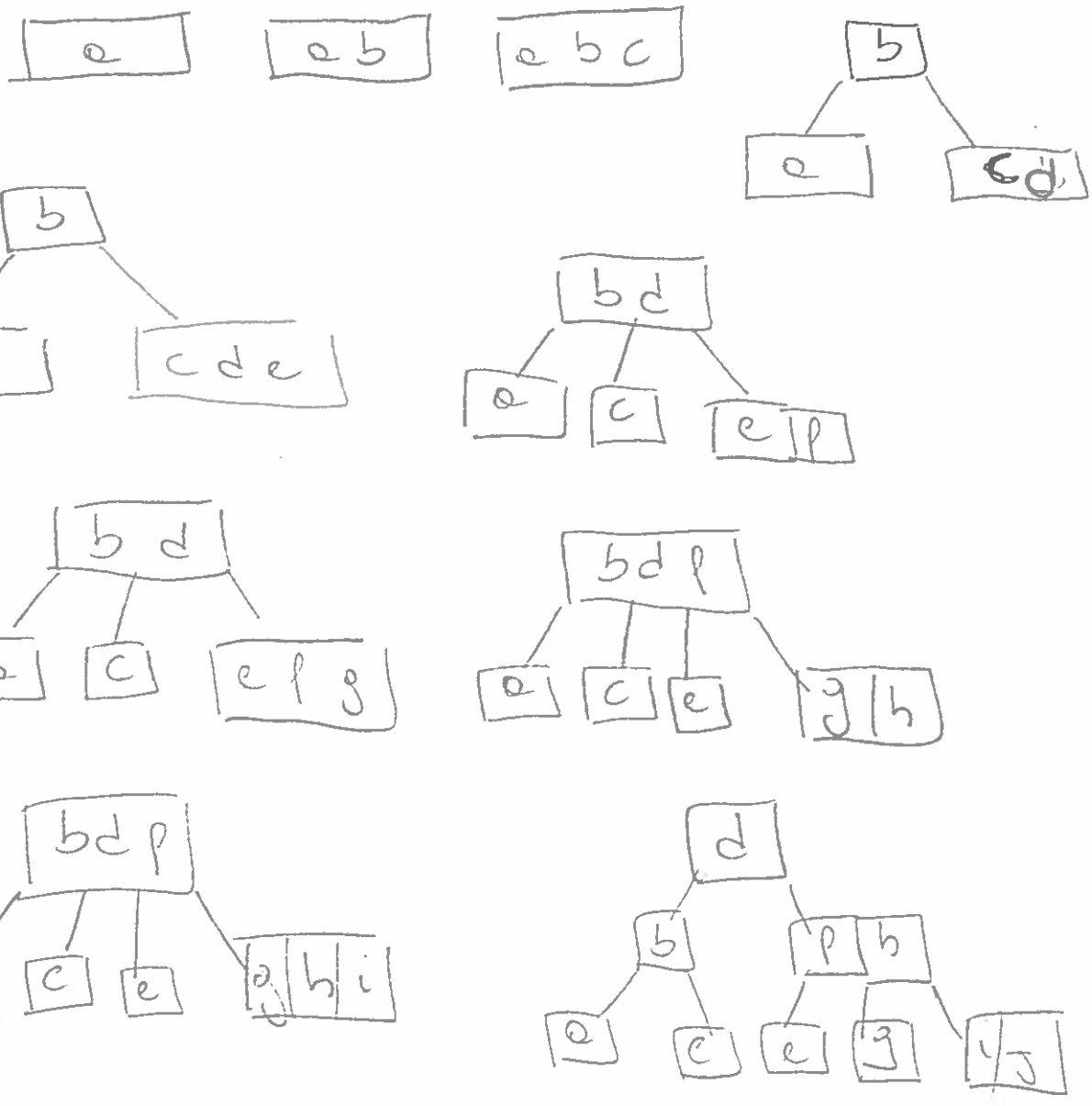
  z = 8 substitute z with its successor



Every node has #keys in $[t-1, 2t-1]$
Root #keys in $[1, 2t-1]$

- **Q6(20 points): B Trees**

  (a)(10 points) For a sequence of keys {a, b, c, d, e, f, g, h, i, j}, suppose we would like to construct a B-Tree, with degree 2, by successively inserting those keys, one at a time, into an initially empty tree. Please draw the sequence of B-Trees after inserting each of the 10 keys.
  Note: please draw only one tree after each insertion. $2t-1 = 3$



(b)(5 points) Let t be the (minimal) degree of a BTree. Suppose the size of each object, including the key, stored in the tree is 40 bytes. Also, suppose the size of a BTreeNode pointer is 4 bytes. In addition, 100 bytes of meta-data is required for each BTree node to keep track of some useful information. Suppose each BTreeNode has only the meta-data, a parent pointer, a list of objects, and a list of child pointers. What is the optimal (minimal) degree for this BTree if a disk block is 4096 bytes?

For each node I am storing (at maximum)

$2t-1$ objs (40 bytes)

$2t$ pointers (4 bytes)

100 bytes metadata

+ 1 pointer to the parent (4 bytes)

$(2t-1)40 + (2t+1)4 + 100 = 4096$

$80t - 40 + 8t + 4 + 100 = 4096$

$88t + 64 = 4096$

$t = \left\lfloor \frac{4096 - 64}{88} \right\rfloor = 45$

(c)(5 points) For a BTree with height 4 (or 5 levels), what is the maximal number of objects can be stored if the (minimal) degree $t = 51$?

$$\#\text{nodes} = \sum_{i=0}^{h} (2t)^i$$

$\forall$ node we have $(2t-1)$ keys

$$(2t-1) \sum_{i=0}^{h} (2t)^i = (2t-1) \frac{(2t)^{h+1} - 1}{2t-1} = (2t)^{h+1} - 1$$

$$(2 \cdot 51)^5 - 1 = 102^5 - 1 =$$