Algorithms Report2 (Due date: 5 PM, Oct. 28)

I) Problem solving manually

(Keys are considered from left to right for the insertion, deletion and etc.)
(Must write down the problem solving process.)

1. Consider inserting the keys 31, 11, 20, 6, 21, 15, 7, 27, 14, 13, 4, 2 into a hash table of length m = 5 using separate chaining where $h(k) = k \bmod m$. Draw the resulting hash tables after inserting all these keys.

2. Consider inserting the keys 12, 9, 8, 17, 2, 10, 31, 1, 18 into a hash table of length **m =11** using open addressing with the auxiliary hash function $h'(k) = k \bmod m$. Draw the resulting hash tables after inserting all these keys.

a) using linear probing whith $h(k,i) = (h'(k) + i) \bmod m$

b) using quadratic probing with $h(k,i) = (h'(k) + c_1 + c_2 i^2) \bmod m$,
where $c_1 = 1$ and $c_2 = 3$

c) using double hashing with $h(k,i) = (h'(k) + i h_2(k)) \bmod m$,
where $h_2(k) = 1 + (k \bmod (m-1))$,

for i=0,1,...,m-1. (Must show the all the hash calculations.)

3. Using Figure 7.1 as a model, illustrate the operation of PARTITION in quicksort on the arrays shown below. The pivot is the first element in A.

a) A = <11, 19, 9, 5, 12, 8, 17, 14, 1, 2, 6, 15>

b) A = <9, 8, 7, 6, 5, 4, 3, 2, 1>.

4. Answer the following questions for the keys

9, 20, 7, 6, 16, 3, 19, 8, 15, 14, 5

a) Draw the final structure of binary search tree T when above keys are inserted from left to right.

b) Draw the tree that results after successively executing the following functions. TREE-DELETE(T,19), TREE-DELETE(T,16), TREE-DELETE(T,7), TREE-DELETE(T,9).
Draw the tree each time TREE-DELETE is executed.

5. Write pseudo code for TREE-PREDECESSOR procedure

6. Draw the red-black tree that results after TREE-INSERT is called on the tree in Figure 13.1(c) with key 5. If the inserted node is colored red, is the resulting tree a red-black tree? What if it is colored black? Answer without TREE-INSERT-FIXUP execution.

7. Draw the red-black trees that result from successive insertions of the keys in the order 11, 8, 12, 5, 13, 2, 10, 17, 26, 9 into an initially empty red-black tree.
   For each insertion, count the number of color changes, left rotations, and right rotations. Also, count the sum of each of these three operations respectively.

8. Draw the red-black trees that result from the successive deletions of the keys in the order 9, 12, 10, 8, 11, 5, 10 on the tree generated in exercise 7.
   For each deletion, count the number of color changes, left rotations, and right rotations. Also, count the sum of each of these three operations respectively.

9. Suppose that a node x is inserted into a red-black tree with RB-INSERT and then it is immediately deleted with RB-DELETE. Is the resulting red-black tree the same as the initial red-black tree? Justify your answer.

II) Programming

1. Hash Table (Separate Chaining)

● Construct the hash table with separate chaining according to the following direction.

1) Hash functions (Construct three hash tables.)

a) h(k) = k mod 5
b) h(k) = k mod 7
c) h(k) = k mod 13

2) Insert the 50 keys to above three tables.
The keys are generated by rand()%1000. Execute srand(time(NULL)) first.
(Duplicate keys are ignored, that is, avoid identical values when randomly generating values.)
Implement a function that inserts the randomly generated keys into the hash table.

3) Print the three tables.
(Different chains in a table should be printed in different lines.)

- Implement a print function for a hash table printing.
- Input to the function is a hash table.
- The print function should print the shortest, longest, and average length of the chains for each hash table.

2. Construct the open address hash table according to the following description.

- m = 37
- Hash functions

  linear probing: $h(k, i) = (h'(k) + i) \bmod m$

  where, $h'(k) = k \bmod m$

  quadratic probing: $h(k, i) = (h'(k) + c_1 i + c_2 i^2) \bmod m$

  where, $h'(k) = k \bmod m$, $c_1 = 1$, $c_2 = 3$.

  double hashing: $h(k, i) = (h_1(k) + i h_2(k)) \bmod m$,

  where, $h_1(k) = k \bmod m$, $h_2(k) = 1 + (k \bmod (m - 1))$.

- 30 **keys** that are randomly generated.

  The keys are generated by rand()%1000. Execute srand(time(NULL)) first.

  (Duplicate keys are ignored, that is, avoid identical values when randomly generating values.)

1) Insert 30 randomly generated keys per hash table into 3 separate hash tables.
   (note that you need a function for 90 insertions, 30 for each table)

2) Print the contents of the hash table for above three different hash functions.
   (Must show the correct positions of the inserted keys in the table).
   The print function also should print the average number of probes per insertion up to now and the primary (largest) cluster length.

3. RBT (Red-Black Tree)

1) Program the following functions.

a. RB-INSERT(T, k) /* **Do not insert k (key value) if it is already in T (RBT)** */

b. RB-DELETE(T, k)

c. PRINT-BST(T) /* Prints the structure of a red-black tree in a way that the structure is visually identifiable (similar to the way when you draw the tree by hand). It is not for just printing the content of the tree. How to distinguish the nodes and edges, the color of the nodes, etc. are all up to you to determine. */

2) Using RB-INSERT(T, k) construct a RBT with the keys in A[15] and print the T.

   Fill in A[15] by rand()%50. Execute srand(time(NULL)) first.

   (Duplicate keys are ignored, that is, avoid identical values when randomly generating values.)

3) Execute RB-INSERT(T, 13), RB-INSERT(T, 22), RB-INSERT(T, 21),
RB-INSERT(T, 11), RB-INSERT(T, 45), sequentially.
Execute PRINT-BST(T), each time after the insertion is performed.
(If a key is already in T the insertion is ignored.)

4) Execute NEAREST-NEIGHBOR(T, 5), NEAREST-NEIGHBOR(T, 23),
NEAREST-NEIGHBOR(T, 38) sequentially.
What is the return value of each execution?
Also, check whether the return values are correct by looking at the structure of the RBT - RBT formed after executing problem 3).

5) Execute RB-DELETE(T, 13), RB-DELETE(T, 22), RB-DELETE(T, 45),
RB-DELETE(T, 11), RB-DELETE(T, root-key) sequentially.
/* where the root-key indicates the current root-key value of RBT */
Execute PRINT-BST(T) each time after the deletion is performed.

How to submit the report.

▶ Need to upload the report2 in a zip file in the i-campus.
Refer to the manual file for uploading in the i-campus.

▶ The zip file should contain the following three files.
1) Document file (.hwp, photo, or scan): Problem solving manually part.
2) C program file(s) (source program): Programming part.
3) Test result file(s): Contains all the screen copy of the test results.

▶ The zip file should be named as shown below,
report2_id_name.zip
example) report2_2020123456_HongGilDong.zip
or report2_2020123456_홍길동.zip

The zip file contains above 1), 2), and 3).

▶ Use windows OS and visual studio program.