115 marks **CS 213(M) Quiz 2** **11:00-13:00, 27/2/19**

Problem 1:(a)[10 marks] Suppose the numbers 1, 2, 3, 4, 5, 6, 7 are inserted into an initially empty binary search tree in some order. (i) Which numbers if inserted in step 1, will imply that the final result tree will be not balanced? (ii) Which numbers if inserted in step 3 will imply that the final result tree will not be balanced? (iii) Which numbers if inserted in step 7 will imply that the the final result will not be balanced?

(b)[10 marks] In the code below which pointers are dangling pointers? Which pointers are such that the memory they point to will leak? Be precise: if `v` is of type `A*`, then state clearly whether the memory for `*v` leaks or the memory for `*(v->x)`. Very little explanation, if any, is expected.

```
struct A{
  int *x;
  A(){ x = new int[10]; }
};

A* f(A* &u){
  A *p = new A, *q = new A, r;
  u = p;
  return &r;
}

int main(){ A *s, *t; t = f(s); delete t; delete s;}
```

Problem 2: A vector (from math) is *sparse* if only a small number of its elements are non-zero. In such a case it is useful to keep track of indices which correspond to the non-zero elements, and the values of those non-zero elements. In particular, the zero elements are not explicitly stored. It is natural to represent a sparse math vector using a `map` as follows.

```
map<int,double> v; // all elements considered 0.
v[10] = 3.5;       // v[10] becomes 3.5, others considered 0.
```

(a)[5 marks] Write a function `void print(map<int,double> v, int n)` which will print elements 0 through `n-1` on the screen. Note that a 0 should be printed for elements which were not explicitly assigned.

(b)[15 marks] Write a function `double dot(map<int, double> v, map<int,double> w)` which returns the dot product of the math vectors v, w. In other words, it should return $\sum_i v[i]w[i]$ where i ranges over all valid indices. Minimize the work done by your function.

(c)[5 marks] A matrix is likewise considered sparse if it contains mostly 0s. Suppose we wish to represent sparse matrices in a manner similar to above. Further, to assign 3.14 to `i,j`th entry of matrix `A` we would like to write `A[i][j] = 3.14`. What would the declaration of `A` be?

Problem 3: Consider a vector class which has indexing, the usual `push_back` operation and a new `pop_back` operation, which causes the last element to be removed from the vector. Here is a possible implementation. At each point during the execution we will have allocated an array of some M elements. Of these, some $N \leq M$ will be in use, to store the current vector. If we perform a `push_back`, and the required length $N+1$ becomes bigger than M, we allocate a fresh array of size $2N$. We copy the vector elements into the newly allocated array, and delete the old array. If we perform a `pop_back` operation, and if the new vector length $N-1$ becomes smaller than $M/3$, then we allocate an array of size $N-1$, copy the vector into the new array, and delete the old array. And of course we update N, M appropriately.

Suppose some n operations are performed including creation, `push_back`, `pop_back` and indexing.

(a) [10 marks] Suppose the rth operation among these is a `pop_back` and causes memory allocation and copying of Q elements. Show that for some constant c, the cQ operations preceding the rth operation do not involve memory allocation.

(b) [10 marks] Show something similar in case the rth operation is a `push_back`.

(c) [5 marks] Using the preceding two parts, show that the total work for the n operations must be $O(n)$.

Problem 4:[20 marks] Suppose we have a binary search tree, in which each node has members `value, left, right` as usual, but in addition there is a member `size` which gives the number of nodes in the subtree of the node (including the node itself). Write functions to (a) insert a value into the tree, and (b) to determine the number of values in the tree smaller than a given x.

Problem 5:(a)[10 marks] What is the minimum number of keys that a 2-3 tree must have so that if a key is inserted its height increases to 3? Note that a tree of height 3 has 4 levels of vertices. Draw the picture of such a tree. Your tree must have distinct positive integers and you should minimize the largest integer in the tree. State what you insert and also draw the result.

(b)[5 marks] Suppose I have a 2-3 tree of height h. What is the minimum number of nodes it can have?

(c)[10 marks] What is the minimum number of keys that can be inserted into a tree of part (b) so that its height increases to $h+1$? Answer for $h = 0, 1, 2, 3$ and see if you can infer a formula. You dont have to prove the formula, enough if it matches $h = 0, 1, 2, 3$.