CPSC 221: Sample Exam Problems

Last Updated: December 3, 2008

Due: NOT TO BE TURNED IN.

Note: There aren't many complexity or counting problems in here, because you already have plenty of those. Also don't imagine that the number of questions of a particular type indicates anything about the number of such questions on the final exam.

Pointers and Dynamic Data

1. There are 3 errors in the following code fragment. List each erroneous statement and briefly describe the error in it.

```
const int NUM = 777;
int main(void) {
  int x = 10;
  int *p = NUM, *q = 0;

  *p = x;
  x = *q;
  &x = NUM;
}
```

2. Find 3 errors in the following function and give brief an explanation for each error.

```
int* foo( int& n) {
  int a[10];

  for (int i = 0; i <= 10; i++)
    a[i] = i;
    if (0 <= n && n <= 10)
      return a + n;
    return NULL;
}
```

3. Consider the following statements:

```
int x = 5;
int y = 3;
int *p, *q;
p = &x;
q = &y;
y = *p;
p = q;
*p *= *q;
```

Draw a diagram showing the values of `x`, `y`, `p`, and `q`, *after* the execution of the above statements.

4. Write a set of statements that creates a *dynamic* array A of 100 integers and sets their values to 0.

5. Suppose `SomeClass` is defined as follows:

```
class SomeClass {
  private:
    string * sa;
    int * ia;
    int size;
  public:
    SomeClass( int n ) {
      sa = new string[n];
      ia = new int[n];
      size = n;
    };

    ~SomeClass();
};
```

where `string` is the class of strings you're familiar with from the lectures, assignments, and tutorials. Write the code for the *destructor* of this class. (Omit this question, if destructors were not covered in the lectures/tutorials.)

Linked Structures

1. Suppose the nodes of a linked list structure are defined as following:

```
struct node {
  int value;
  node* next;
};
```

Define a function `length` that takes as argument a linked list and returns the number of items that are in it. The function must leave its argument unchanged.

For instance, if `L` is the list `[3,9,5,6]` then `length(L)` returns 4 and `L` is unchanged.

2. Suppose the nodes of a doubly linked list structure are defined as:

```
struct node {
  int item;
  node* next;
  node* prev;
};
```

Write a function `concat` that concatenates two given lists (the first node of the second list will follow the last node of the first list) and returns the new list. Note that `concat` does not create new nodes; it just rearranges the links of some existing notes.

```
node* concat( node* list1, node* list2 ) {
```

3. Suppose the nodes of a doubly linked list structure are defined as in the previous question. Define a function `add_ordered` that takes a new item and a list whose items are kept in increasing order and inserts the new item in the right place in the list (so the items in the list are in increasing order after the insertion).

   For instance, if `L` is the list `[3,5,6,9]` then after `add_ordered(L,8)`, `L` becomes `[3,5,6,8,9]`.

4. Suppose the nodes of a binary tree structure are defined as follows:

```
struct node {
  int value;
  node* left;
  node* right;
};
```

   Define a function `count` that takes as argument a pointer to the root node of a binary tree and returns the number of nodes that are in the tree. The function must leave the tree unchanged.

Data Structures, etc.

1. Suppose that we use a linked list to represent a queue and that in addition to the enqueue() and dequeue() functions (i.e., functions to add and remove elements from the linked list), you want to add a new operation to the queue that deletes the last element of the queue. Which linked structure do we need to use to guarantee that this operation is also executed in constant time? Justify your answer.

2. Suppose the nodes of a binary tree structure are defined as:

```
struct Bnode {
  int value;
  Bnode* left;
  Bnode* right;
}
```

   Define a function `height` that takes the address of a node in a binary tree and returns the height of the node in that tree. Recall: The height of a node in a tree is the length of the longest path from the node to a leaf. The height of a node without children is 0, and for this function, if the user passes in a null node, you can assume that its height is 0, too. Here's the function header to help you get started.

```
int height( Bnode* node) {
```

3. What is the worst-case running time for inserting n items into an initially empty hash table, where collisions are resolved by chaining using a singly-linked list? What if each linked list must be in sorted order?

4. Suppose that each row of an $n \times n$ array A consists of 1's and 0's such that, in any row of A, all the 1's come before any 0's in that row. Assuming A is already in memory, describe a function running in $O(n \log n)$ time (not $\Theta(n^2)$ time) for counting the number of 1's in A.

5. Let A be a collection of objects. Describe (in words) an efficient function for converting A into a set. That is, remove all duplicates from A. What is the running time of this method?

Data Structures and Graph Theory

1. Consider 6-tuples of the form (bandName, rating, lyrics, lifestyle, generosity, friendship) where bandName is a string, and the other 5 attributes are integers having values ranging from 1-100 (1=bad, 100=good).

   We want to know the asymptotic running time of the following two algorithms:

   Algorithm A

```
u = 0
Open the ROCK-tree
While (data exists in the update file) do
   Read a 6-tuple from the update file
   u++
   Try to insert the 6-tuple into the ROCK-tree
   If (insert failed for any reason) then
      print "insert failed for tuple ", u
   else
      print "insert was successful for tuple ", u
```

   Algorithm B

```
// Perform a range query to see which bands are most similar to a
// given band (i.e., the closest in terms of the 5 numeric
// attributes for the band).
Print "Enter the name of a band to search for its neighbours"
Read targetBandName
Print "Enter a radius for the range search (e.g.,10, 500, 25000)"
Read radius
Search the ROCK-tree to find all other bands that lie within a
hypersphere of radius units of the query band. Specifically, return:
   (a) head = a pointer to a list of bands (i.e., a pointer to a list of 6-tuples)
   (b) n = the number of bands falling within the radius
Print out all the tuples in the results list
```

The following information was taken from the ROCK-tree programmer's manual:

(a) It takes $O(1)$ time to open any file, including the ROCK-tree.

(b) A ROCK-tree can store k-tuples of any size k. Smaller k means faster searches. A k-tuple consists of k attributes—namely 1 unique string, and $k-1$ integers. All tuples must have the same k.

(c) It takes $O(k \log n)$ time to insert a k-tuple into a ROCK-tree containing n tuples.

(d) Given a unique string (e.g., band name), it takes $O(\log n)$ time to locate a target k-tuple in the ROCK-tree containing n tuples, and then $O(kd^2 + x)$ expected time to create a list of all x k-tuples that lie within radius d of that target k-tuple.

(e) It takes $O(k)$ time to print a k-tuple.

(f) You can assume that the update file is much smaller in size than the data that is currently contained in the ROCK-tree.

Determine the worst-case time complexity of **Algorithm A**, and the expected time complexity of **Algorithm B**. Explain your reasoning.

2. Consider the same 6-tuples from the previous question, assuming a band name length of 30 bytes, and a pointer length of 10 bytes. This time, instead of using a ROCK-tree, we'll use a B^+-tree. Answer the following questions, and be sure to show your work.

   (a) Estimate the number of these data entries that can be stored in a B^+-tree leaf node of size 4K, assuming we're only storing key-pointer pairs.

   (b) Estimate the minimum amount of space (total number of B^+-tree nodes) that you would need to store 10,000 bands in the index.

   (c) Suppose that we place the whole record/tuple (band name and all of its five 4-byte integer ratings) directly in the index. In other words, instead of having key-pointer fields, we have key-record fields. Now repeat parts (a) and (b) with this change. You can assume that the key is part of the record, so you don't have to count space for it twice.

3. A rock band wants to perform in the following cities: Vancouver, Surrey, Burnaby, Los Angeles, San Francisco, New York, and Toronto. Assume every city has an airport (except Burnaby and Surrey) that has direct flights to every other airport. The band lives in Vancouver so it has decided to perform at some (perhaps none) of Vancouver, Surrey, and Burnaby before going on tour and to perform at the rest (perhaps none) when it gets back.

   (a) How many possible orderings are there for the band to perform in all 7 of these cities, assuming they perform in each city exactly once?

   (b) Suppose the band wants to visit each of the four non-local cities and Vancouver exactly once, returning to Vancouver (i.e., they fly in, then they give their concert in the city, then they get questioned at police headquarters about all kinds of things, and then they head out to the airport to leave the city—being warned not to return again). Is such a cycle possible among the five cities? If so, give one such ordering. If not, explain why no such ordering is possible.

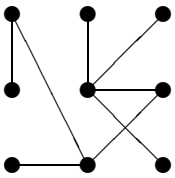
(c) Can the band visit all seven cities exactly once assuming that to get anywhere non-local to/from Vancouver, Surrey, or Burnaby the band must fly to/from Vancouver? If so, give one such ordering. If not, explain why no such ordering is possible.

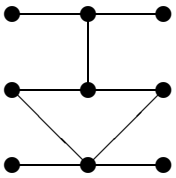
4. After completing their North American tour in (3), the band members notice that the more they fly, the more frequent flyer points they get. So, they decide that they want to fly on all of the flights that Euler Airlines offers between all the following cities. They plan to travel first class, get free beverages on board each flight, mingle with the flight staff, and collect their frequent flyer points. Euler Airlines has flights between these cities. A "1" means that a single one-way flight exists from the source city (row) to the destination city (column).

	Van	Tor	LA	NY	SF
Van	0	0	1	0	1
Tor	0	0	1	1	0
LA	1	1	0	1	1
NY	0	1	1	0	0
SF	1	0	1	0	0

Assuming the legal authorities actually permit the band to travel to these cities, is there an Euler cycle that the band can complete using each of the one-way flights in the table? If so, give one such cycle. If an Euler cycle is not possible, then: (a) state why not, and (b) determine whether an Euler path is possible (giving an example if there is one, or an explanation if there isn't one).

5. A rock band has time to kill between flights, and wants to know how to turn the following graphs into trees.

(a) 

(b) 

Re-draw, if possible, each graph as an m-ary (maximum m children per parent) rooted tree, where $m = 3$. Also, re-draw, if possible, each graph as a balanced tree (where "balanced" means that all of the leaves appear on no more than 2 adjacent levels: level x and (possibly) level $x - 1$, where x is the height of the tree.

6. Let G be a simple connected graph with n vertices and m edges. Explain why $\log m$ is $O(\log n)$.

7. Use Euler's formula to show: If $G = (V, E)$ is a simple connected planar graph with $|V| \geq 3$ then $|E| \leq 3|V| - 6$.