CS 216 Exam 1 – Spring 2001

Part 1 – Closed Book

Name _______________________  Section _______________________

Email Address _________________  Student ID # _________________

Pledge:

This exam is in two parts. Once you are finished with part 1, **hand it in**, and you will be given part 2. Part 1 is worth 70 points and Part 2 is worth 30 points. You will have an hour and fifty minutes total to complete both parts.

**Part 1:** Closed note, closed book. You are not to speak with anyone except the Instructor or a teaching assistant for any reason except an emergency during the exam. No calculators are allowed.

**Part 2:** Open book, open notes. You may not speak with anyone except the Instructor or a teaching assistant for any reason except an emergency during the exam. Calculators are allowed. You may not borrow notes or textbooks from other students.

Obviously, you are not to discuss the exam with anyone until after everyone has taken it! (Tuesday at 5:30pm)

Good Luck!

<table>
<thead>
<tr>
<th>PART</th>
<th>MAX</th>
<th>SCORE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>70</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>TOTAL</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>
1. (8 points) Define the terms. Be as specific as possible.

(a) binary search tree

(b) depth of a node in a tree

(c) AVL property

(d) mantissa

2. a) (2 points) How many bits are required to represent integers in the range –256 to 255 in two’s complement notation?

b) (2 points) Name two reasons why two’s complement representation is preferable over sign magnitude.
3. (9 points) What is the representation of each of the following in the indicated radix? Be sure to show your work.

(a) 361_8 in hex

(b) 1IA_{19} in decimal

(c) 109_{10} in radix 6

4. (6 points) Consider the positive binary integer 0100110101111101_2. Express
   
   a. this binary number in octal
   
   b. this binary number in hexadecimal
   
   c. the negative of the number, assuming a two’s complement representation (same number of bits)
5. (12 points total) Describe the running time of the following pseudocode in Big O notation:

a) for (int i = 0; i < n; ++i) {
    for (int j = 0; j < n; ++j) {
        a = a + b;
    }
}

b) for (int i = 0; i < 10000; ++i) {
    for (int j = 0; j < n; ++j) {
        a = a + b;
        b = b + c;
        cout << a + b;
    }
}

c) for (int i = 0; i < n; ++i) {
    for (int j = 0; j < n; ++j) {
        a = b + c;
    }
    for (int k = 0; k < n; ++k) {
        cout << a + b;
    }
}

d) p = 0;
    while (p < n) {
        A[p] = new int;
        p++;
    }
6. (4 points) Consider a binary search tree that is built by inserting the sequence:
   \[5, 1, 7, 6, 8, 4, 2, 3\]
   into an initially empty binary tree.

   Draw the tree constructed by this insertion sequence.

7. (7 points) Given the following tree:

   \[
   \begin{array}{c}
   \ \ \ \ \ 7 \\
   1 \quad 8 \\
   2 \quad 9 \\
   \end{array}
   \]

   a) Is it an AVL tree? If not, circle the nodes that violate the AVL property. Why or why not (must answer for any credit)?

   b) List the order in which the nodes are visited for a postorder traversal:
8. (5 points)
a) Describe 2 different methods of implementing the size() function on a linked list class.

b) For each method, describe a (general) situation where that method would be preferable to the other method.

9. (3 points) Draw the BST tree after removing the value 6 from the tree. Use the method we described in class.

```
       6
      / \
     1   10
    /     \
   2       7
        /   \
       8     11
            /   \
           9     12
```
10. (6 points) I am writing a program that will keep track of student grades for cs216. I will use a student record that holds the name, student number, and grades for each student. Give me one reason I might want to use each of the following data structures AND one reason I might not want to use this structure:

Array of records –

Linked List of records –

Binary Search Tree of records –

11. (6 points) The 216vector class is implemented using arrays, however, there is no limit on the number of elements you can have in a 216vector. The way we accomplish this is by initially allocating an array of some size, and then when we need more space, we re-allocate another array of double the size of the current one. We then copy the elements of the old 216vector into the new 216vector and then delete the old 216vector.

What will be the worst case running time in big-O notation of each of the following operations? Describe how you came up with that value.

a) Pushback (add an element at the end of the 216vector) –

b) operator[] to access an element –

c) Insert_before(int loc, element val) – (inserts val before location loc) –
12. (12 points) Given a 16 bit word machine with IEEE floating point format depicted as:

\[
\begin{array}{cccccc}
\text{Bits} & 0 & 1 & \ldots & 5 & 6 & \ldots & \ldots & 15 \\
\text{sign} & \text{exp} & \text{mantissa}
\end{array}
\]

Assume that the mantissa is 10 bits (11 with hidden) and that the exponent is 5 bits with excess 8 format. What is the representation (in binary and in hex) of the exponent in the representation for

a) \(23.0_{10}\)

b) \(3.625 \times 2^{-5}\)
13. (18 points) Show how you would implement a queue using two stacks.

a) (4 points) Write the general idea of how your enqueue and dequeue functions would work.

b) (8 points) Given the .h file for the Queue class, and the .h file for a Stack class as described in Weiss on p. 94, write C++ code for the enqueue and dequeue methods. The .h file for the Stack class is reproduced on the next page for your convenience.

Add any extra data members to the Queue class if needed for your implementation.

c) (6 points) What is the big-O running time of your enqueue function? Your dequeue function? Explain briefly how you got your answer.
// Stack Implemented as a linked list. Reproduced here for your
// convenience. From p. 94 in Weiss.

template <class Object>
class Stack
{
    public:
    Stack( );
    Stack( const Stack & rhs );
    ~Stack( );

    bool isEmpty( ) const;
    bool isFull( ) const;
    const Object & top( ) const;

    void makeEmpty( );
    void pop( );
    void push( const Object & x );
    Object topAndPop( );

    const Stack & operator=( const Stack & rhs );

    private:
    struct ListNode
    {
        Object element;
        ListNode *next;

        ListNode( const Object & theElement,
                  ListNode * n = NULL )
            : element( theElement ), next( n ) { }
    };

    ListNode *topOfStack;
};
/ This is the new class, a queue implemented with two
// stacks: (based on the Queue class interface on p. 113 of
// Weiss) Implement the enqueue and dequeue methods on the next
// page. Add any additional private data members (if any)
// necessary for your implementation.

template <class Object>
class Queue
{
    public:
        Queue( int capacity = 10 );

        bool isEmpty( ) const;
        bool isFull( ) const;
        const Object & getFront( ) const;

        void makeEmpty( );
        Object dequeue( );
        void enqueue( const Object & x );

    private:

        Stack<Object> Stack1;
        Stack<Object> Stack2;

        // Add other data members as needed:

        void increment( int & x );
};
Write your code for the enqueue and dequeue methods on this page. If you need to add any more private data members to the Queue class, add them on the previous page.