CS 216 Exam 1 – Fall 2003

Name:_______________________  Section:_______________________

Email Address:_________________  Student ID #_______________________

This exam is closed note, closed book. You will have an hour and fifty minutes total to complete the exam. You may NOT use calculators.

Good Luck!

Write and sign pledge after taking the exam:
Define each term and give an example that explains it:

1. (2 points) Abstract Data Type

2. (2 points) Hidden Bit in IEEE Floating Point

3. (2 points) LIFO data structure

4. (2 points) Height of a tree
5. (2 points) Describe a situation when a big-Oh comparison is useful.

6. (2 points) Describe a situation when a big-Oh comparison is NOT useful.

7. (6 points) The Virginia DMV has asked you to create a data structure that will keep track of license plate patterns. A license plate can have 8 characters, and each character can be a capital letter, a digit, a dash, a space, or an ampersand (&).

   a) How many bits will be required to represent a single character?

   b) In addition, a plate can be one of 180 different patterns. How many bits does it take to represent which pattern the customer has selected?

   c) Using your answer from part a and part b, how many bytes does it take to specify a customer’s license plate?
8. (4 points) Given the following infix expression: \(((d / ((a + g) * (b - e)) )^c)\)

   a) Write it as a prefix expression:

   \[ d / (a + g) * (b - e) \]

   b) Write it as a postfix expression:

   \[ d a g + b e - * / \]
9. (10 points total) Describe the running time of the following pseudocode in Big-Oh notation in terms of the variable \( n \). Assume all variables used have been declared.

*Show your work for partial credit.*

```c
int foo(int k) {
    int sum;
    if (k > 1) {
        for (int i = 0; i < k; ++i) {
            sum += i;
        }
    }
    return sum;
}

int bar(int k) {
    if (k < -10)
        return 0;
    else if (k < 10)
        return (2 * bar(k-2));
    else
        return (1 + bar(k-1));
}
```

a) answ = foo(n);
b) answ = bar(n);

c) for (int j = 0; j < n; ++j) {
    cin >> val;
    for (int i = 0; i < j; ++i) {
        b = b * val;
        for (int k = 0; k < n; ++k)
            c = b + c;
    }
}
d) for (int i = 0; i < n * n; ++i) {
    if (i < 100) {
        for (int j = 0; j < i; ++j)
            j >> cout;
    }
}
e) b = 1000;
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < b; ++j){
            sum = sum + i;
        }
        for (int k = 0; k < n; ++k){
            a[k] = a[k] + sum;
        }
    }
10. (6 points total) What is the representation of each of the following in the indicated radix? Be sure to show your work.

a) \(177_{11}\) in decimal

b) \(2313_4\) in hex

c) \(12G_{18}\) in radix 10

11. (6 points total) Consider the positive binary integer represented in two’s complement: \(0001110110010010_2\).

a) Express this binary number in octal

b. Express this binary number in hexadecimal

c. Negate the number (i.e. give the two’s complement representation of a negative version of the same number) Use the same number of bits.
12. (3 points) Draw the binary search tree created by inserting these values in this order:

5 4 3 2 8 6 0 1 9 7

13. (2 points) Give a pre-order traversal of your tree shown above:

14. (2 points) Give a post-order traversal of your tree shown above:

15. (3 points) Delete the root of the tree shown above using one of the two methods described in class. Draw the new tree here:
16. For each operation below give: 1) How you would most efficiently implement the operation, 2) Describe the worst case scenario (e.g. “The worst case occurs when the value you are looking for is not in the list”) and 3) What is the worst case Big-Oh running time of this scenario. State any assumptions you make.

a) (3 points) Push a value onto a stack implemented as an array.

b) (3 points) Print all the values in an AVL tree (BST) from smallest to largest.

c) (3 points) Find the minimum value in a list implemented as an array.
17. (3 points) Given the following tree:

```
  32
 /   \
24    44
 /     \
7      60
 /   \
6      31
```

Is it an AVL tree? If not, circle the node(s) where the AVL property is violated. **Why or why not** (must answer for any credit)?

18. (3 points) Given the following AVL tree:

```
  8
 / \
5   9
/   \
3   6
```

Insert the value 7 into the AVL tree above, doing any necessary rotations to maintain the AVL property.
19. (3 points) Given the following AVL tree:

```
    7
   / \
  4   8
 /   /    \
3   5       8
```

Insert the value 2 into the AVL tree above, doing any necessary rotations to maintain the AVL property.
20. (7 points) Assume we are using the 32-bit IEEE single precision floating point format as described in class and used in lab. The mantissa has 24 bits including the hidden bit, there is one sign bit, and there are eight exponent bits. The exponent is stored in excess 127.

What decimal floating point number is represented by the following 32 bits? SHOW YOUR WORK!

0101 0111 0110 1000 0000 0000 0000 0000

a) Is this a positive or negative number?

b) What is the exponent (in base 10)?

c) What is the value of the mantissa (in base 10)

d) What is the total value?

Note: you may leave your answer in the form: \( \text{value}_{10} \times \text{base}^{\text{exponent}} \)

Where you specify value, base and exponent.
21. (20 points) This question tests your understanding of stacks and pointer manipulation. You must implement a stack ADT in C++. The underlying representation of the stack should be the Node class as described below. Your stack should store integers and should handle errors (printing an error message is fine).

You will be graded mostly on the correctness of the ideas of your solution rather than exact C++ syntax, but your solution should be clear. Correct C++ code is the best way to ensure we understand your solution. You may NOT use the STL in any way for this question.

You should implement all the functions with \( \rightarrow \) in front of them.

```cpp
class Node {
public:
    Node(int value) {val=value; prev = next = NULL;};
    int val;
    Node *prev;
    Node *next;
};

class Stack {
public:
    \( \rightarrow \) Stack(); // constructor
    \( \rightarrow \) void push(int value); // pushes value onto the stack.
    \( \rightarrow \) int pop();   // returns and removes the value on
    \( \rightarrow \) int top();   // returns the value on the top of
    \( \rightarrow \) bool isEmpty();   // returns true if the stack contains
    \( \rightarrow \) // the top of the stack.
    \( \rightarrow \) // the stack without removing it.
    \( \rightarrow \) // no elements.
private:
    // Add data members here.
};
```
[extra space for use in previous question]
22. (8 points) Many stack calculators, such as HP's scientific calculator line, implement stack manipulation operations. These include:

\[
\begin{align*}
n \text{ rotate} \\
n \text{ roll}
\end{align*}
\]

Rotate has the following semantics: Remove the bottom n items from the stack and place them on the top of the stack, without otherwise changing the order of any elements.

Roll removes the nth item on the stack and places it at the top, without otherwise changing order in the stack.

Assume you have a binary version of the stack from the previous question (#21), and a definition of the interface. Implement non-member functions to implement rotate and roll on a stack.
[ Extra space for previous problem]