ECE 250
Data Structures and Algorithms
QUIZ 2

2006-10-16

The quiz is out of 20 marks.
No questions, no aides.
If you are unsure about a question, write down your assumptions and continue.
This examination has two pages of questions.
If you run out of room, use the reverse of this page.

Surname, Given Name				**Student ID**	
1.	**2.**	**3.**	**4.**	**5.**	**B.**

_____

Sign here to indicate that you have read the above instructions.

1. **[3]** Discuss the appropriateness of using a queue in a client-server model with respect to the following topics:

a. speed (quantitative answer using asymptotic analysis),
b. implementation and maintenance (qualitative answer), and
c. fairness in satisfying the requests of the clients (qualitative answer).

Use **one** reasonable-length sentence for answering each of parts a, b, and c.

2. **[3]** Finish the following proof-by-induction:

To show that $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$, we first see that $\sum_{i=1}^{1} i = 1 = \frac{1(1+1)}{2}$.

Next, we assume that the statement is true for $n = k$, that is, we assume $\sum_{i=1}^{k} i = \frac{k(k+1)}{2}$.

Show that, using this assumption, the statement is true for $n = k + 1$.

3. **[6]** Fill in Table 3 for the three given nodes in Table 3. Give the height of the sub-tree with that node as its root and give the depth relative to the entire tree. Leave a blank if there are no appropriate entries.

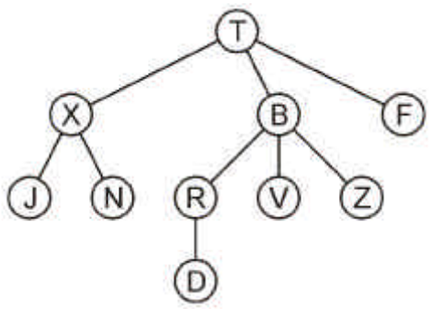

Figure 1. A general tree.

Table 3.

Node	Parent	Siblings	Children	Height	Depth
T					
B					
N					

4. **[2]** What is the state of the stack after having used the stack to evaluate the following partial expression given in reverse-Polish notation? Three tables are provided for your answer and rough work. Please indicate which table holds your final answer.

3 5 + 4 7 * 2 4 + + 5

Table 4a.

Table 4b.

Table 4c.

5. **[6]** Implement (write a definition for) the member function **`void swap()`** which swaps the first two elements in a singly-linked list as implemented in Project 1. Throw an **`underflow()`** exception if there are not at least two objects in the list and leave the list unchanged. For reference, the class member declaration is:

```
template <typename Object>
class SingleList {
        private:
                SingleNode<Object> * list_head;
                SingleNode<Object> * list_tail;
                int count;

        public:
                // constructor/copy constructor/destructor/operator =

                // Accessors

                int size() const;
                bool empty() const;

                Object front() const;
                Object back() const;

                SingleNode<Object> * head() const;
                SingleNode<Object> * tail() const;

                bool member( const Object & obj ) const;

                // Mutators

                void push_front( const Object & obj );
                void push_back( const Object & obj );

                Object pop_front();
                void swap();

                bool remove( const Object & obj );
};
```

Note: there are **many** possible solutions. So long as the final state of the linked list is consistent, you are welcome to use whatever approach you wish so long as the list is left in the correct state.