MIDTERM EXAMINATION

SEMESTER FALL 2003

CS301-DATA STRUCTURE

Total Marks:86

Duration: 60min

Instructions

Please read the following instructions carefully before attempting any question:

1. **The duration of this examination is 60 Mins.**
2. **This examination is closed book, closed notes, closed neighbors; any one found cheating will get no grade.**
3. **Unless stated otherwise, all questions carry a single mark.**
4. **Do not ask any questions about the contents of this examination from anyone.**
   a. **If you think that there is something wrong w ith any of the questions, attempt it to the best of your understanding.**
   b. **If you believe that some essential piece of information is missing, make an appropriate assumption and use it to solve the problem.**
5. **Most, but not all, of the examination consists of multiple-choice questions. Choose only one choice as your answ er.**
   a. **If you believe that two (or more) of the choices are the correct ones for a particular question, choose the best one.**
   b. **On the other hand, if you believe that all of the choices provided for a particular question are the wrong ones, select the one that appears to you as being the least wrong.**

7. **You are allowed to use any development environment like Dev C++ etc.**

Question No: 1 **Marks: 2**

Here is the start of a class declaration:

```
class Foo
{
public:
void x(Foo f);
                    void y(const Foo f);
void z(Foo f) const;
...
```

Which of the three member functions can change the *PRIVATE* member variables of the Foo object that activates the function?

a. Only x and y
b. Only x and z
c. Only y and z
d. None of three the functions.
e. All three functions.

Question No: 2 **Marks: 2**

What is the common pattern of writing class definitions?

a. Member functions and member variables are both private.
b. Member functions are private, and member variables are public.
c. Member functions are public, and member variables are private.
d. Member functions and member variables are both public.

Question No: 3 **Marks: 2**

The Bag ADT is like the List ADT. The Bag ADT does not store items in any particular order and it allows duplicates. Suppose that the Bag class is efficiently implemented with a fixed array with a capacity of 4000. Insert appends the new item at the end of the array. Choose the best description of b's member variables size (count of items in the bag) and data (the array that holds the actual items) after we execute these statements:

```
Bag b;
b.insert(5);
b.insert(4);
b.insert(6);
```

What will be the values of b.size and b.data after the statements?

a. b.size is 3, b.data[0] is 4, b.data[1] is 5, b.data[2] is 6
b. b.size is 3, b.data[0] is 5, b.data[1] is 4, b.data[2] is 6
c. b.size is 3, b.data[0] is 6, b.data[1] is 4, b.data[2] is 5
d. b.size is 3, b.data[0] is 6, b.data[1] is 5, b.data[2] is 4

Question No: 4 **Marks: 2**

The operation for adding an entry to a stack is traditionally called:

a. add
b. append
c. insert
d. push

Question No: 5 **Marks: 5**

Consider the following pseudo code:

```
declare a stack of characters
while ( there are more characters in
the word to read ) {
      read a character
      push the character on the stack
}
while ( the stack is not empty ) {
            pop a character off the stack
            write the character to the screen
}
```

What is written to the screen for the input “ carpets ”?

a. serc
b. carpets
c. steprac
d. ccaarrppeettss

Question No: 6 **Marks: 2**

In the linked list implementation of the stack class, where does the push member function place the new entry on the linked list?

a. At the head
b. At the tail
c. After all other entries that are greater than the new entry.
d. After all other entries that are smaller than the new entry.

Question No: 7 **Marks: 2**

One difference between a queue and a stack is:

a. Queues require dynamic memory, but stacks do not.
b. Stacks require dynamic memory, but queues do not.
c. Queues use two ends of the structure; stacks use only one.
d. Stacks use two ends of the structure, queues use only one.

Question No: 8 **Marks: 2**

I have implemented the queue with a linked list, keeping track of a front pointer and a rear pointer. Which of these pointers will change during an insertion into a *NONEMPTY* queue?

a. Neither changes
b. Only front pointer changes.
c. Only rear pointer changes.
d. Both change.

Question No: 9 **Marks: 2**

I have implemented the queue with a linked list, keeping track of a front pointer and a rear pointer. Which of these pointers will change during an insertion into an *EMPTY* queue?

a. Neither changes
b. Only front pointer changes.
c. Only rear pointer changes.
d. Both change.

Question No: 10 **Marks: 2**

In a single function declaration, what is the maximum number of statements that may be recursive calls?

a. 1
b. 2
c. n (where n is the argument)
d. There is no fixed maximum

Question No: 11 **Marks: 2**

What is the maximum depth of recursive calls a function may make?

a. 1
b. 2
c. n (where n is the argument)
d. There is no fixed maximum

Question No: 12 **Marks: 2**

In which location do dynamic variables reside?

a. The code segment.
b. The data segment.
c. The heap.
d. The run-time stack

Question No: 13 **Marks: 6**

For public part of the Throttle declaration below, mark each function member header as follows:

- **Mark C for any constructor;**
- **mark X for any function that is forbidden from changing the throttles data fields.**

```
class Throttle
{
public:
        Throttle( );
        Throttle(int size);
        void shut_off( );
        void shift(int amount);
        double flow( ) const;
        bool is_on( ) const;
        ...
```

Answer/Solution

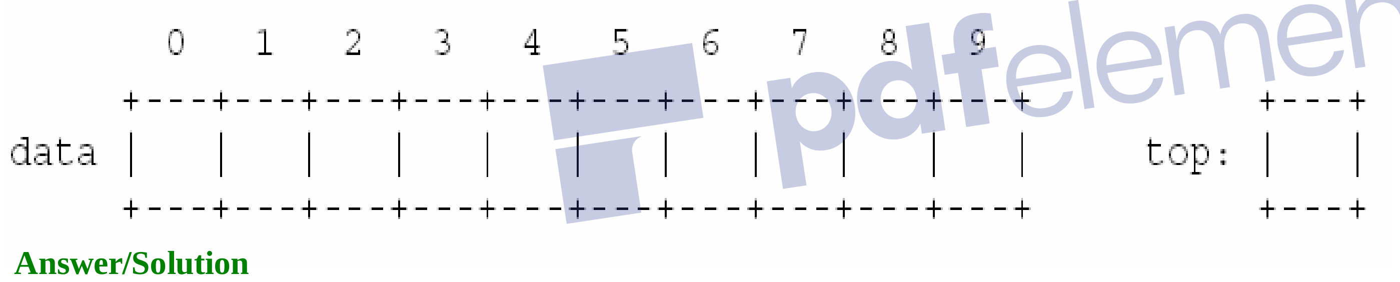
```
class Throttle
{
public:
        Throttle( );                    C
        Throttle(int size);             C
        void shut_off( );
        void shift(int amount);
        double flow( ) const;           X
        bool is_on( ) const;       X
        ...
```

Question No: 14 **Marks: 5**

I am going to execute this code with THREE pushes and ONE pop:

```
Stack s;
s.push(1);
s.push(2);
s.push(3);
cout << s.pop( );
```

Suppose that the stack s is represented by *a fixed-sized array*. Draw the state of the private member variables " data " and "top " of "s " after the above code:



```
       0   1   2   3   4   5   6   7   8   9
     +---+---+---+---+---+---+---+---+---+---+          +---+
data |   |   |   |   |   |   |   |   |   |   |     top: |   |
     +---+---+---+---+---+---+---+---+---+---+          +---+
```

Answer/Solution

```
0 1 2 3 4 5 6 7 8 9
1 2          Top1
```

Question No: 15 **Marks: 10**

Complete the body of this function. Use a Queue of characters to store the input line as it is being read.

```
int counter( )
// Precondition:
// There is a line of input waiting to be read from cin.
// Postcondition:
// A line of input has been read from cin, up to but not
// including the newline character. The return value of
// the function is the number of times that the LAST
// character of the line appeared somewhere in this line.

// EXAMPLE
// Input: ABBXDXXZX The value returned by counter would
//        be 4 for this input since there are 4 X's in
//        the input line.
           {
           int answer = 0;
           Queue q;
```

Answer/Solution

```
int counter()
{
char a[100];
int i=0;
int answer=0;
Queue q;

cin.getline(a,98,'\n');
for(i=0;i<strlen(a);i++)
{
q.enqueue(a[i]);
}

i--;
while(!q.isEmpty())
{
if(a[i]==q.dequeue())
{
answer++;
}
}
return answer;
}
```

Question No: 16 **Marks: 5**

I am going to execute this code with THREE insert s and ONE remove :

```
Queue s;
s.insert(1);
s.insert(2);
s.insert(3);
cout << s.remove();
```

Suppose that s is represented by a singly linked linked list. Draw the linked list and the state of the private member variables of s after the above code:

front_ptr

rear_ptr

Answer/Solution

2 3

rear_prt

Front_prt

Question No: 17 **Marks: 10**

Consider CList and Node classes defined as follows:

```
class Node

{
    public:
    Node *next;
    Node *prev;
    int data;
};
class CList
{
    public:
    void insertHead(int);
    void insertTail(int);
    void removeHead();
    void removeTail();
    bool isEmpty();
    bool find(int);
    private:
    Node *head;
    Node *tail;
};
```

A. write the body of the member function insertHead which inserts a new element at the head of the list.

```
void Clist::insertHead( int x )
{
```

B. write the body of the member function removeTail which removes the element at the tail of the list.

```
void Clist::removeTail( int x )
{
```

Answer/Solution

(a) Solution for Question 17 option (a)

```
void   CList::insertHead(                int   x)
{
Node *newNode=               new Node();
newNode->data=x;
newNode->next=NULL;
newNode->prev=NULL;
          if  (isEmpty())
head=tail=newNode;
          else
{
newNode->next=head;
newNode->prev=NULL;
head->prev=newNode;
head=newNode;
}

}
```

(b) Solution for Question 17 option (b)

```
void   CList::removeTail(                int   &x)
{
          if  (isEmpty())
                    return     ;
          else
{
Node *p=tail;
                    if  (head==tail)
head=tail=NULL;
                    else
{
tail=tail->prev;
tail->next=NULL;
}
x=p->data;
                    delete     p;
                    return     ;
}
               }
```

Question No: 18 **Marks: 10**

Trace the running of the infix to postfix conversion algorithm on the infix expression
(A - B) + C/D

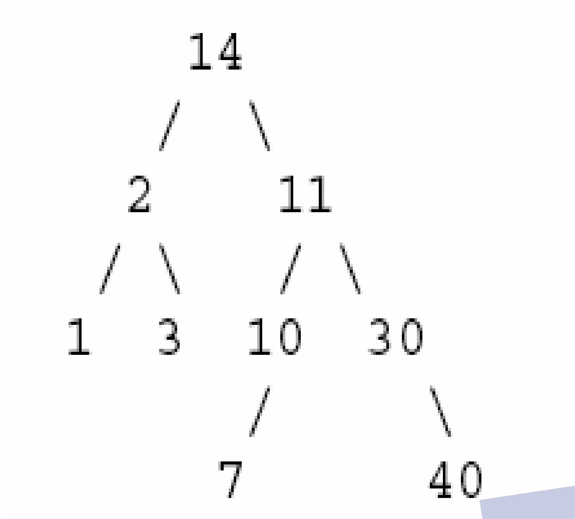
Answer/Solution

Symbol Postfix Stack
((
A A (
- A (-
B AB (-

) AB- (
AB-
+ AB- +
C AB-C +
/ AB-C +/

Question No: 19 **Marks: 13**

Here is a small binary tree:



A. What are all the leaves? (2pts)
C. What are the ancestors of the node 10? (2pts)
D. What are the descendants of the node 30? (2pts)
E. Is the tree a binary search tree (BST) (true/false)? (2pts)
F. Print the tree when visited in post-order manner? (5pts)

Answer/Solution

A) Leaves of the tree = 1,3,7,40
B) Ancestors of the node 10 = 11,14
C) Descendants of the node 30 = 40
D) Is the tree a binary search tree (BST) (true/false) False
E) Post Order Traversal = 1,3,2,7,10,40,30,11,14