Data Abstractions and Structures

PRACTICE Midterm Exam #1

Pseudo-code, Big O Notation, Linear Lists, Stacks, and Queues

60min, 100 points, open book, open notes.

1. (15 points) – Pseudocode Algorithm Efficiency and the Big O Notation
   Read pseudocode and explain what it does
   Read pseudocode and determine the Big O Notation

2. (20 points) – Singly, Doubly and Multi-linked lists
   Doubly-Linked list – basic operations: insert, delete, search, traverse, etc.
   Multi-linked list – basic operations: insert, delete, search, traverse, etc.
   Header and Sentinel nodes

3. (20 points) – Stacks
   Stack Applications – infix, postfix, prefix, evaluate postfix
   Stack Operations
   Stack ADT

4. (15 points) – Queues
   Basic queue operations: enqueue, dequeue, etc.
   Queue applications
   Queue ADT

5. (30 points)

Stacks & Queues: Write a function or pseudocode for problems such as

A. Write a reverse stack function to be added to the stack class
B. Write a reverse stack function calling existing stack functions (and using other temp stacks as needed).
C. Given a stack and a queue, write a function that calls existing stack/queue functions to check if they contain the same data (top of the stack must be identical to the front of the queue, etc.)

OR

Singly and Doubly-Linked Lists with sentinel node[s]: Write a function or pseudocode for problems such as

A. Swap consecutive nodes in a doubly-linked list.
B. Swap any two nodes in a doubly-linked list. See a calling statement below

```
done = list.swap(3, 6);
done = list.swap(6, 2);
```

1. (15 Points)

```
// A. (3 Points)
// What is the BigO?
// What is the output?

n = 16
i = 2
loop( i < n )
   print (i)
   i = i + 2
end loop
```

```
// B. (3 Points)
// What is the BigO?
// What is the output?

n = 32
i = 1
loop( i <= n )
   print (i)
   i = i * 2
end loop
```

```
// B. (4 Points)
// What is the BigO?
// How many '*' will be displayed?

n = 10
i = 0
loop( i < n )
   j = 0
   loop( j < n )
      k = 0
      loop ( k < n )
          print ("*")
          k = k + 1
      end loop
      j = j + 1
   end loop
   i = i + 1
end loop
```

```
// D. (5 Points)
// An array has n = 8 numbers listed below:
// 7, 4, 10, 5, 10, 6, 9, 8
// What does the guess function return?
// What is the BigO?

Algorithm guess(ary, n )
Pre:      ary – has data
          n – its actual size
Post:

   k = 0
   i = 1
   loop( i < n )
      if( ary[k] > ary[i] )
         k = i
      end if
      i = i + 1
   end loop
   return k
end guess
```

2. (20 Points) Imagine we have a doubly-linked list with two sentinel nodes. Assume that the list contains: **hdS 10 20 30 40 50 60 70 tlS**. The linked list node has three fields: **data**, an integer, and two pointers named **forw** and **back**. The list class has three data members: a pointer to the first node, named **head**, a pointer to the last node, named **tail**, and a counter named **cnt**.

A. (4 Points) What is the output?

```
cout << head->forw->data;
cout << tail->back->data;
```

B. (4 Points) What is the output?

```
cout << tail->back->back->data;
cout << tail->back->back->back->forw->data;
```

C. (6 Points) The following code checks if a list is sorted in ascending order. It does not work properly. Find and correct the errors.

```
Algorithm testSorted()

      pCurr = head
      loop (pCurr != tail)
          if (pCurr->data < pCurr->forw->data)
               return true
          else
               return false
      end loop

end testSorted
```

D. (6 Points) Write code that prints the node(s) in the middle of the list. For instance, if the list contains: **hdS 10 20 30 40 50 60 70 tlS**, the output should be: **40**
if the list contains: **hdS 10 20 30 40 50 60 tlS**, the output should be: **30 40**

3. (20 Points)

A. (5 Points) What would the contents of **s1** and **s2** be after the following code is executed? Input sequence: **5, 0, 0, 0, -2, 0, 0, 4, 0, 0, 0, -3**

```
loop( not end of input)
   read( data )
   if( data != 0 )
       s1.push(data)
   end if
end loop
loop(s1 not empty )
   s1.getTop( item )
   if( item > 0 )
       s2.push(item)
   end if
   s1.pop(data)
   s2.push(data)
end loop
```

B. (5 Points) Give the prefix and postfix forms of the following expression:

A - B + C % (D - 2)

C. (5 Points) Evaluate the following expression using a stack:

100 5 - 2 2 * 5 3 * + * 2 -

D. (5 Points) The following algorithm fragment rearranges the nodes in a stack so that the last node is moved in the beginning of the stack without changing the other nodes. For instance, if the stack contains (from **top** to base): **5 6 7 8**, after processing it should contain: **8 5 6 7**. It does not work properly. Find and correct the

```
pre = NULL
curr = top
loop ( curr not NULL )
    curr = curr->next
    pre = curr
end loop
top = pre
pre->next = NULL
curr->next = top
```

4. (15 Points)

A. (3 Points) For each of the following situations which of the ADTs would be most appropriate? (1) –a stack, (2) – a queue, (3) – none of these

a. Integers that need to be sorted
b. Airplanes that stack above an airport, waiting to land
c. Customers at a deli counter who take numbers to mark their turn

B. (3 Points) An operation that displays the contents of a queue can be useful during program debugging. The algorithm below does not work properly. Explain why.

```
loop(que not empty)
     que.dequeue(item)
     print ( item )
end loop
```

C. (4 Points) If the numbers **10 20 30** are placed in a queue (in that order), what is the contents of the queue after executing the following pseudo-code?

```
que.getFront(num1)
que.getRear(num2)
num3 = num1 + num2
que.enqueue(num3)
que.dequeue(num3)
que.enqueue(num3)
```

D. (5 Points) The following algorithm removes zeros from a queue of integers. It does not work properly. Find and correct the errors

```
loop(que not empty)
     que.dequeue(item)
     if ( item != 0 )
         que.enqueue(item)
     end if
end loop
```

5. (30 `Points`) Imagine we have a doubly-linked list with two sentinel nodes. The linked list node has three fields: **`data`**, an integer, and two pointers named **`forw`** and **`back`**. The list class has three data members: a pointer to the first node, named **`head`**, a pointer to the last node, named **`tail`**, and a counter named **`cnt`**. Write a member function that swaps any two data nodes in the. See a calling statement below

```
        done = list.swap(3, 6);
        done = list.swap(6, 2);
```