CSE 225

Homework 1

Question 1

Implement a sorted linked list data structure in which each node contains an integer value. Write a function to insert a new element into the sorted linked list while maintaining its sorted order.

Instructions:

1. Define a structure for a node in the sorted linked list. Each node should contain an integer value and a pointer to the next node.
2. Implement a function to insert a new element into the sorted linked list while keeping the list sorted in ascending order.
3. Ensure that duplicate elements are allowed in the sorted linked list.
4. Write a function to print the elements of the sorted linked list in order.

Your task is to write functions to achieve the following:

1. **insert_sorted**: This function should take a pointer to the head of the sorted linked list and an integer value to be inserted. It should insert the value into the appropriate position in the sorted linked list while maintaining the sorted order.
2. **print_sorted_list**: This function should print the elements of the list in ascending order.

Question 2

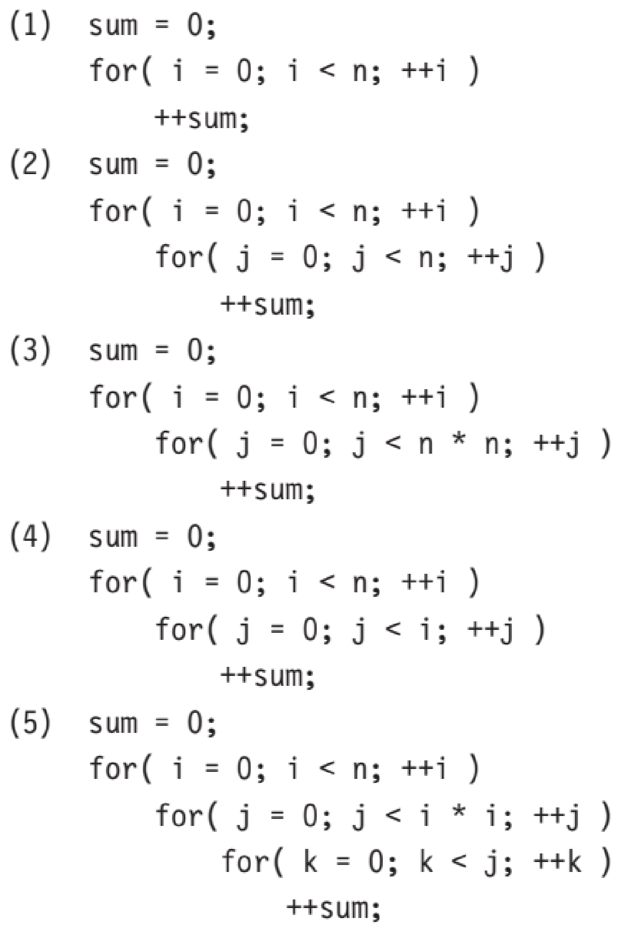
(2.1) Analyze the running time of the function *InsertItem* shown below (sorted list). To get credit, you need to be as specific as possible.

```
template <class ItemType>
void SortedType<ItemType>::InsertItem(ItemType item)
{
 int location = 0;
 bool found;

 found = false;
 while( (location < length) && !found) {

   if(item > info[location])
     location++;
   else
     found = true;

 }

 for(int index = length; index > location; index--)
   info[index] = info[index - 1];
 info[location] = item;
 length++;
}
```

(2.2) What are the main differences between static and dynamic array allocation?

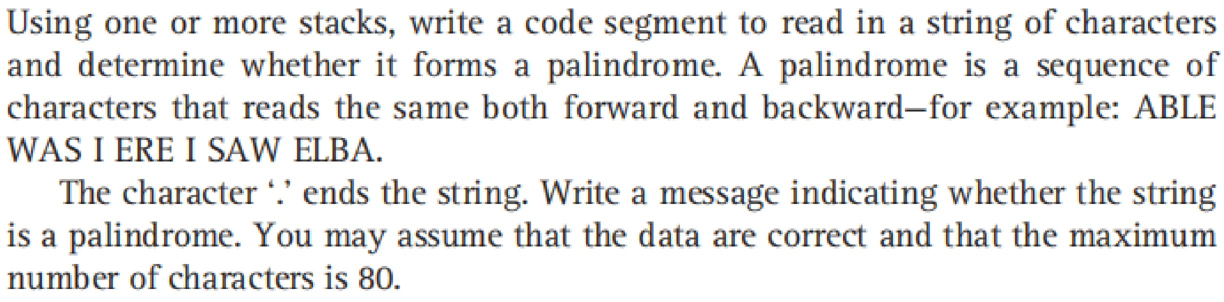
Question 3

For each of the following program fragments give an analysis of the running time using Big-Oh



```
(1)  sum = 0;
     for( i = 0; i < n; ++i )
         ++sum;
(2)  sum = 0;
     for( i = 0; i < n; ++i )
         for( j = 0; j < n; ++j )
             ++sum;
(3)  sum = 0;
     for( i = 0; i < n; ++i )
         for( j = 0; j < n * n; ++j )
             ++sum;
(4)  sum = 0;
     for( i = 0; i < n; ++i )
         for( j = 0; j < i; ++j )
             ++sum;
(5)  sum = 0;
     for( i = 0; i < n; ++i )
         for( j = 0; j < i * i; ++j )
             for( k = 0; k < j; ++k )
                 ++sum;
```

Question 4



Using one or more stacks, write a code segment to read in a string of characters and determine whether it forms a palindrome. A palindrome is a sequence of characters that reads the same both forward and backward—for example: ABLE WAS I ERE I SAW ELBA.

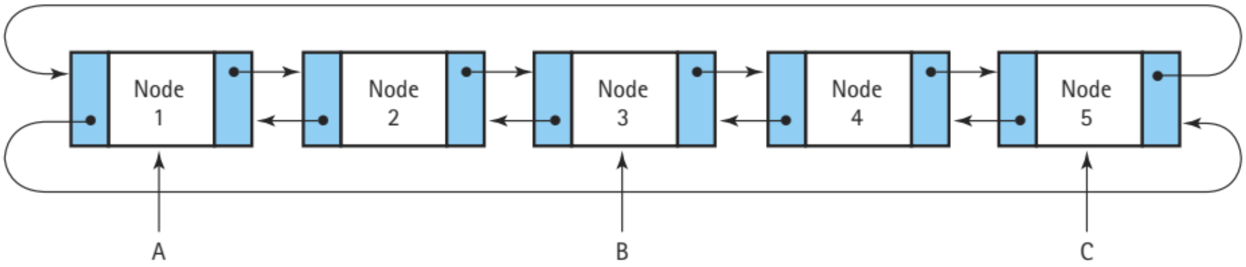
The character '.' ends the string. Write a message indicating whether the string is a palindrome. You may assume that the data are correct and that the maximum number of characters is 80.

Question 5

Write the C++ code for evaluating a postfix expression (The algorithm was discussed in Stack and Queue Slide).

Question 6

Using the circular doubly linked list below, give the expression corresponding to each of the following descriptions.



(For example, the expression for the **info** member of Node 1, referenced from pointer A, would be **A->info**.)

a. The `info` member of Node 1, referenced from pointer C

b. The `info` member of Node 2, referenced from pointer B

c. The `next` member of Node 2, referenced from pointer A

d. The `next` member of Node 4, referenced from pointer C

e. Node 1, referenced from pointer B

f. The `back` member of Node 4, referenced from pointer C

g. The `back` member of Node 1, referenced from pointer A

Question 7

Two stacks of the same type are the same if they have the same number of elements and their elements at the corresponding positions are the same. Overload the relational operator **==** for the **class** **stackType** that returns **true** if two stacks of the same type are the same; it returns **false** otherwise. Also, write the definition of the function template to overload this operator.

Question 8

Given a queue of integers, write an algorithm that, using only the queue ADT, calculates and prints the sum and the average of the integers in the queue without changing the contents of the queue.