Exam 1

Exam information

CS 250 , Fall 2018

Name: _____

Exam Format: This exam covers C++ Review, STL Structures, Arrays, Pointers, Memory Management, Exceptions, Linked Lists, and Algorithm Efficiency concepts.

Each question can receive between 0 and 4 points, and each question has a weight associated with it. The point value is used to compute the score for a question. For example, if a question is worth a weight of 5% and the student receives 3 points, then that question will count for 3.75% out of the full 5%.

0	1	2	3	4
Nothing written	Attempted, but incorrect	Partially correct; multiple errors	Mostly correct, one or two errors	Perfect; correct answer & notation

#	Question	Weight	Points Received
1	STL Structures	8%	
2	Graph identification	12%	
3	Pointers	12%	
4	Random/Sequential access	8%	
5	Linked List diagrams	24%	
6	Coding - dynamic variables/arrays	10%	
7	Coding - node	11%	
8	Coding - linked list	15%	

Scratch paper

Part 1: Theory

10% Question 1: STL Structures □ 0 □ 1 □ 2 □ 3 □ 4

Circle the correct answer for each.

a. An STL vector is implemented as a (choose one)
○ linked list ○ dynamic array ○ normal array ○ queue

b. And the STL list is implemented as a (choose one)
○ linked list ○ dynamic array ○ normal array ○ queue

10% Question 2: Identification □ 0 □ 1 □ 2 □ 3 □ 4

Label each of the given structures – what data type does each figure represent?

Figure 1

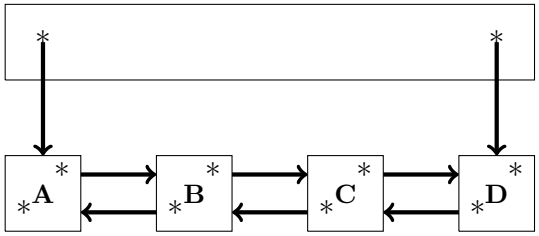
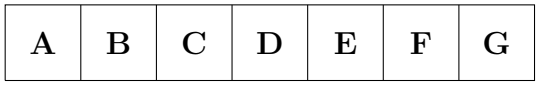


Figure 2



10% Question 3: Pointers □ 0 □ 1 □ 2 □ 3 □ 4

a. When a pointer is not currently in use, it should be set to what?

What is the reason for this?

b. If not initialized by the programmer, a pointer has a default value of what?

10% Question 4: Random access/sequential access □ 0 □ 1 □ 2 □ 3 □ 4

Which of the following structures allows for **random access** of its elements?

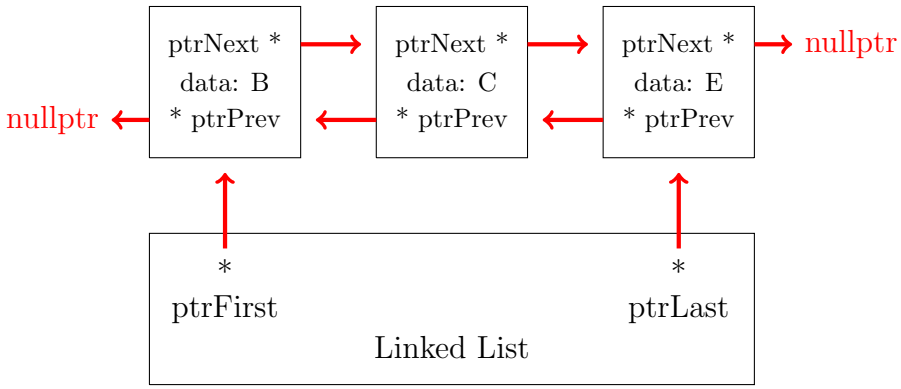
○ Link-based Structure ○ Array-based Structure

Which of the following structures requires **traversal** over all elements 0 through $n-1$ to get to index n?

○ Link-based Structure ○ Array-based Structure

10% Question 5: Linked List Diagram □ 0 □ 1 □ 2 □ 3 □ 4

Given this diagram as the starting point:



Answer each question. Each question should add on to each previous diagram. Make sure to always draw out the Linked List and all pointers.

a. Draw the diagram after a `Insert("D")` occurs, assuming the list will always be in alphabetical order. (A, then B, then C, ...)

b. Draw the diagram after a `PushFront("A")` occurs.

c. Draw the diagram after a `PopBack()` occurs.

Part 2: Coding

10% Question 6: Dynamic variable and array □ 0 □ 1 □ 2 □ 3 □ 4

a. Complete the following code for a dynamic variable:

```
\\ Create pointer:
Node * n;


 \\ Allocate space for a variable via n:




 \\ Free space pointed to by n:
```

b. Complete the following code for a dynamic array:

```
\\ Create pointer:
int * arr;


 \\ Allocate space for an array, of size 20, via arr:




 \\ Free space pointer to by arr:
```

10% Question 7: Linked List Node □ 0 □ 1 □ 2 □ 3 □ 4

Write out the class declaration for a Node that would be used by a DoublyLinkedList. Include the data (any data type) and any appropriate pointer variables.

```
class Node {

};
```

30% Question 8: Coding □ 0 □ 1 □ 2 □ 3 □ 4

Write out C++ code or pseudocode for the following Linked List functions. Use the Node that you declared in the previous question.

a. PushFront

```
void DoublyLinkedList::PushFront( TYPE newData ) {

}
```

b. PopBack

```
void DoublyLinkedList::PopBack() {

}
```

c. GetAtIndex (aka operator[ ])

```
TYPE DoublyLinkedList::GetAtIndex( int index ) {

}
```