QUESTION 1

3 points ✓ Saved

If we have a LinkedList L (20; 30; 40).
Using the LinkedList specifications from the slides, how can we insert 10 at the start of the list?

```
L.findFirst();
Integer temp = L.retrieve();
L.insert(new Integer(10));
L.update(temp);
---------------------------
```

```
L.findFirst();
L.insert(new Integer(10));
---------------------------
```

```
L.findFirst();
Integer temp = L.retrieve();
L.update(new Integer(10));
L.insert(temp);
---------------------------
```

```
L.findFirst();
L.insert(L.retrieve());
L.update(new Integer(10));
---------------------------
```

QUESTION 2

2 points ✓ Saved

What is the output of the following code

```
LinkedList<Integer> A = new LinkedList<Integer>();
A.insert(new Integer(100));
A.insert(new Integer(66));
A.insert(new Integer(13));
A.insert(new Integer(15));
A.findfirst();
A.insert(new Integer(34));
A.insert(new Integer(56));
A.findnext();
A.insert(new Integer(24));
while(!A.last()) {
 System.out.print(A.retrieve() + ", ");
 A.findnext();
}
```

- ○ 13,
- ○ 100, 34, 56, 66, 24, 13, 15,
- ○ 24, 13, 15,
- ◉ 24, 13,
- ○ 100, 66, 13, 15, 34, 56, 24,

QUESTION 3

2 points ✓ Saved

As a User of the ADT List, we created a removeAll method that removes all the elements of a LinkedList.

```
public static <T> void removeAll(LinkedList<T> A) {
            A.findfirst();
            while(! A.last())
                  A.remove();
}
```

How good will it do?

- ○ It will not work; the parameter should be LinkedList<Integer> instead of LinkedList<T>
- ○ It will work properly and delete all elements
- ○ It will not delete the first element
- ○ It will cause a runtime error
- ◉ It will not delete the last element

QUESTION 4

0 points Saved

What will this code do?

```
A.findfirst();
while(! A.last()) {
  A.findnext();
  A.insert(A.retrieve())
}
```

- ○ It will duplicate all elements except for the 1st and last
- ○ It will duplicate all the List elements
- ◉ The code will duplicate all elements except first one
- ○ It will duplicate all elements except for the last
- ○ It will duplicate 2nd, 4th, 6th, ...etc. elements findNext() will skip odd elements

QUESTION 5

2 points ✓ Saved

If we execute the following code on list L with elements: 5, 3, 1, 4, 6

```
L.findFirst();
while(!L.last()) {
        L.remove();
        L.findNext();

}
```

Then the elements of L will be:

- ○ *L* will be empty
- ○ 5,1,6
- ○ 3,4
- ◉ 3,4,6

QUESTION 6

1 points ✓ Saved

As a user of an ADT List , and with a list L, the result of calling the method “remove()”:

- ○ Removes the current element and sets the new current to the existing successor of the deleted element.
- ○ Removes the current element and sets it to be NULL if the resulting list is empty.
- ○ Removes the current element and sets the new current to be the first element if no successor of the deleted element exists.
- ◉ All of the above.

QUESTION 7

1 points ✔ Saved

As a user of List ADT, the result of calling the method "last()"

- ○ Return the data of the last element.
- ◉ None
- ○ Move Current to be on the last element.
- ○ Return whether the successor of Current is on the last element or not.

QUESTION 8

1 points ✓ Saved

What will this method do to List L?

```
public static <T> void f(List<T> l) {
       if (! l.empty()) {
             l.findFirst();
             int cpt = 1;
             while (!l.last()) {
                   l.insert(l.retrieve());
                   cpt++;
                   l.findNext();
             }
             l.insert(l.retrieve());
             l.findFirst();
             for(int i = 0; i < cpt; i++){
                   l.remove();
                   if (!l.last())
                         l.findNext();
             }
       }
 }
```

- ○ Duplicates all elements of L.
- ○ Infinite loop.
- ◉ None of the above.
- ○ Creates duplicates of all elements of L except the last one.

QUESTION 9

4 points ✓ Saved

Method *findMiddle* in DoubleLinkedList ADT will set the element in the middle of the list as the current element. Fill in the blanks:

```
public void findMiddle() {
      Node<T> temp = head ;
      while(current.next != null)
            current = current. next ;
      while(current != temp) {
            current = current. previous ;
            if(current != temp)
                  temp = temp. next ;
      }
}
```

QUESTION 10

1 points ✔ Saved

As a user of the ADT List, consider the method *inBetween*, that receives a list l and two elements e1 and e2 and returns the number of elements between e1 and e2. Assume that both e1 and e2 exist in the list l, e1 appears before e2 and there are no duplicates. Complete the code below by choosing the correct answer:

```
1. public int inBetween(List l, T e1, T e2){
2.     int count = 0;
3.     ...
4.     while(...) {
5.           ...
6.     }
7.     ...
8.     while(...) {
9.           ...
10.          ...
11.    }
12.    return count;
13. }
```

Line 3:

- ○ `l.findNext();`
- ○ `if(!l.empty())`
- ◉ `l.findFirst();`
- ○ `if(!l.full())`
- ○ None

QUESTION 11

1 points ✓ Saved

As a user of the ADT List, consider the method *inBetween*, that receives a list l and two elements e1 and e2 and returns the number of elements between e1 and e2. Assume that both e1 and e2 exist in the list l, e1 appears before e2 and there are no duplicates. Complete the code below by choosing the correct answer:

```
1. public int inBetween(List l, T e1, T e2){
2.     int count = 0;
3.     ...
4.     while(...) {
5.           ...
6.     }
7.     ...
8.     while(...) {
9.           ...
10.          ...
11.    }
12.    return count;
13. }
```

Line 4:

- ○ `!l.retrieve().equals(e1) && !l.retrieve().equals(e2)`
- ○ `!l.retrieve().equals(e1) || !l.retrieve().equals(e2)`
- ◉ `!l.retrieve().equals(e1)`
- ○ `!l.retrieve().equals(e2)`
- ○ None

QUESTION 12

1 points ✓ Saved

As a user of the ADT List, consider the method *inBetween*, that receives a list l and two elements e1 and e2 and returns the number of elements between e1 and e2. Assume that both e1 and e2 exist in the list l, e1 appears before e2 and there are no duplicates. Complete the code below by choosing the correct answer:

```
1. public int inBetween(List <T> l, T e1, T e2) {
2.     int count = 0;
3.     ...
4.     while(...) {
5.           ...
6.     }
7.     ...
8.     while(...) {
9.           ...
10.          ...
11.    }
12.    return count;
13. }
```

Line 5:

- (●) `l.findNext();`
- ( ) `l.findFirst();`
- ( ) `l.remove();`
- ( ) `count++;`
- ( ) None

QUESTION 13

1 points ✓ Saved

As a user of the ADT List, consider the method *inBetween*, that receives a list l and two elements e1 and e2 and returns the number of elements between e1 and e2. Assume that both e1 and e2 exist in the list l, e1 appears before e2 and there are no duplicates. Complete the code below by choosing the correct answer:

```
1. public int inBetween(List <T> l, T e1, T e2) {
2.     int count = 0;
3.     ...
4.     while(...) {
5.           ...
6.     }
7.     ...
8.     while(...) {
9.           ...
10.          ...
11.    }
12.    return count;
13. }
```

Line 7:

- ○ `l.findFirst();`
- ◉ `l.findNext();`
- ○ `l.insert(e1);`
- ○ `l.insert(e2);`
- ○ None

QUESTION 14

1 points ✓ Saved

As a user of the ADT List, consider the method *inBetween*, that receives a list l and two elements e1 and e2 and returns the number of elements between e1 and e2. Assume that both e1 and e2 exist in the list l, e1 appears before e2 and there are no duplicates. Complete the code below by choosing the correct answer:

```
1. public int inBetween(List <T> l, T e1, T e2) {
2.     int count = 0;
3.     ...
4.     while(...) {
5.           ...
6.     }
7.     ...
8.     while(...) {
9.           ...
10.          ...
11.    }
12.    return count;
13. }
```

Line 8:

- ○ `!l.retrieve().equals(e1) && !l.retrieve().equals(e2)`
- ○ `!l.retrieve().equals(e1) || !l.retrieve().equals(e2)`
- ○ `!l.retrieve().equals(e1)`
- ◉ `!l.retrieve().equals(e2)`
- ○ None

QUESTION 15

1 points ✔ Saved

As a user of the ADT List, consider the method *inBetween*, that receives a list l and two elements e1 and e2 and returns the number of elements between e1 and e2. Assume that both e1 and e2 exist in the list l, e1 appears before e2 and there are no duplicates. Complete the code below by choosing the correct answer:

```
1. public int inBetween(List <T> l, T e1, T e2) {
2.     int count = 0;
3.     ...
4.     while(...) {
5.           ...
6.     }
7.     ...
8.     while(...) {
9.           ...
10.          ...
11.    }
12.    return count;
13. }
```

Line 9:

- (●) `count++;`
- ( ) `l.findNext();`
- ( ) `l.findFirst();`
- ( ) `l.remove();`
- ( ) None

QUESTION 16

1 points ✔ Saved

As a user of the ADT List, consider the method *inBetween*, that receives a list l and two elements e1 and e2 and returns the number of elements between e1 and e2. Assume that both e1 and e2 exist in the list l, e1 appears before e2 and there are no duplicates. Complete the code below by choosing the correct answer:

```
1. public int inBetween(List <T> l, T e1, T e2) {
2.     int count = 0;
3.     ...
4.     while(...) {
5.           ...
6.     }
7.     ...
8.     while(...) {
9.           ...
10.          ...
11.    }
12.    return count;
13. }
```

Line 10:

- ○ `count++;`
- ◉ `l.findNext();`
- ○ `l.findFirst();`
- ○ `l.remove();`
- ○ None