QUESTION 8

2 points Save Answer

What will this code do?

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

- It will duplicate all the List elements
- It will duplicate all elements except for the last
- It will not change the list's original elements
- It will duplicate all elements except for the 1st element
- It will duplicate 2nd, 4th, 6th, ...etc. elements findNext() will skip odd elements

QUESTION 11

1 points Save Answer

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

QUESTION 12

1 points Save Answer

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

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

QUESTION 4

1 points Save Answer

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:

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

QUESTION 13

1 points Save Answer

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 3

1 point

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 4

QUESTION 10

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 2

1 points Save

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 14

1 points Save Answer

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 15

1 points Save Answer

QUESTION 15

1 points Save Answe

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

QUESTION 16

2 points Save Answ

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:

- 3,4,6
- L will be empty
- 5,1,6
- 3,4

QUESTION 1

2 points Save Answer

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();
}
```

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

QUESTION 2

1 points Save Answer

QUESTION 5

What will this method do to DoubleLinkedList L.

```
public static <T> void method1(DoubleLinkedList<T> L)
{
    if (! L.empty())
    {
        DoubleLinkedList<T> R = L;
        while (! R.last())
            R.findNext();
        L.findFirst();

        while(!L.last() && !R.first()){
            T temp = R.retrive();
            R.update(L.retrive());
            L.update(temp);
            L.findNext();
            R.findPrevious();
        }
    }
}
```

- Nothing will change.
- Reverses all L's elements.
- Reverses all L's elements except the last one.
- None of the above.
- Infinite loop.

QUESTION 6

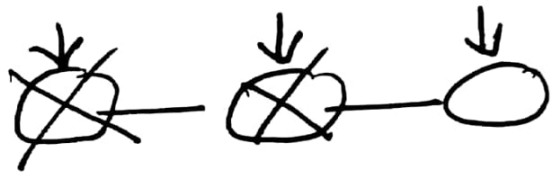
Method *findMiddle* in DoubleLinkedList ADT will set the element in the middle of the list as the current element. Fill in the blanks:

```
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 7

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

QUESTION 9 — 3 points

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.update(new Integer(10));
L.insert(temp);
```

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

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

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