Question1/1

#	Answer
1	**(e)** 2
2	**(a)** $n^2 \log n$
3	**(c)** O(1)
4	**(b)** ArrayList
5	**(a)** O(1)

Question1/2

#	Answer
Line 1	**(d)** 1
Line 2	**(c)** $n \log(n)+1$
Line 3	**(b)** $n^2 \log(n)$
Line 4	**(e)** $(n-1)(n \log(n))$
Line 5	**(e)** 1
Total O	**(a)** $n^2 \log(n)$

Question2/1

```
public static <T> int lastIndex(List<T> l, T e)
{
	int index = -1;
	int i = 0;

	l.findFirst();

	while(! l.last())
	{
		if (l.retrieve().equals(e))
			index = i;

		i++;
		l.findNext();
	}

	if (l.retrieve().equals(e))
		index = i;

	return index;
}
```

Question2/2

```
    public static <T> void reverseCopy(DoubleLinkedList<T> l1,
                                       DoubleLinkedList<T> l2)
    {
         while(! l1.last())
              l1.findNext();

         while(! l1.first())
         {
              l2.insert(l1.retrieve());
              l1.findPrevious();
         }

         l2.insert(l1.retrieve());
    }
```

Question2/2 using Linked List

```
    public static <T> void reverseCopy(LinkedList<T> l1,
                                              LinkedList<T> l2)
    {
         if(! l1.empty())
         {
              l1.findFirst();
              T x;

              l2.insert(l1.retrieve());
              l1.findNext();

              while(! l1.last())
              {
                   x = l2.retrieve();
                   l2.update(l1.retrieve());
                   l2.insert(x);
                   l2.findFirst();

                   l1.findNext();
              }

              x = l2.retrieve();
              l2.update(l1.retrieve());
              l2.insert(x);
              l2.findFirst();
         }
    }
```

Question3/1

```
public void cut(int k)
{
     while(current.next != null)
          current = current.next;

     for(int i = 0 ; i < k; i++)
     {
          if(current.previous != null)
          {
               current.previous.next = null;
               current = current.previous;
          }
          else
               current = head = null;
     }
}
```

Question3/2

```
public void remove(T e)
{
     current = head;

     while(current != null)
     {
          if (current.data.equals(e))
          {
               if (current == head)
                    head = head.next;
               else
               {
                    Node<T> tmp = head;

                    while (tmp.next != current)
                         tmp = tmp.next;

                    tmp.next = current.next;
               }

               if (current.next == null)
                    current = head;
               else
                    current = current.next;
          }
          else
               current = current.next;
     }
}
```

Question about the Queue from Mid2 Spring 2017 - Class memeber

```
    public T serveTail()
    {
        if(size != 0)
        {
            T tmp = tail.data;

            if (size == 1)
                head = tail = null;
            else
            {
                Node<T> prev = head;

                while(prev.next != tail)
                    prev = prev.next;

                prev.next = null;
                tmp = tail.data;
                tail = prev;
            }

            size--;

            return tmp;
        }

        return null;
    }
```

Question about the Queue from Mid2 Spring 2017 – User

```
    public static <T> T serveTail(LinkedQueue<T> q)
    {
        T x;
        T tmp = null;

        for(int i = 0 ; i < q.length() ; i++)
        {
            x = q.serve();

            if (i == q.length())
                tmp = x;
            else
                q.enqueue(x);
        }

        return tmp;
    }
```