* Midterm review
  + 1)
    - String & String :: operator -=(const String & s)
      * Int I = 0;
      * While (I < size)
        + If ( substr(I, s.size) == s)

Shift(I, s.size);

* + - * + Else

I++

* + - * Return \*this;
      * For knowledge heres shift
      * Void String::shift(int start, int len)
        + If (start < 0 || start >= size || len > size)

Return;

* + - * + For (; start + len < size; start++)

Buffer[start] = buffer[start+len];

* + - * + Size -=len;
      * This is O(n^2).
  + 2)
    - There is no push front for the vector because its inefficient. Its an expensive procedure.
    - Template <class T>
    - Void Vector<T>::push\_front(const T & x)
      * Resize(my\_size+1);
      * For(int I = my\_size-1; I > 0; I--)
        + Buffer[I] = buffer[I-1];
      * Buffer[0] = x;
  + 3)
    - Template <class T>
    - Void List<T>:: remove(const T & x)
      * Iterator I = begin();
      * While(I != end())
        + If (\*I == x)

Erase(I);

* + - * + Else

I++;

* + 4)
    - Int main()
      * String word;
      * Int count = 0;
      * Cin >> word;
      * While (!cin.eof())
        + If (word == "the")

Count++;

* + - * + Cin >> word;
      * Cout << count << endl;
* Look at the website. There is homework 3 for merge sort on the site. Due march the 8
* Deque
  + All functions for deque are in the book except the erase (const iterator &)
  + For the iterator class all functions can be in-line. For in-line functions we do not need to comment and explain what each one does.
  + Push front becomes push back on the first vector. As well as front() becomes the back() for the vector one
  + Push back would become push back for vector tw. And back of the deque becomes to back on vector two.
  + If you pop back enough and make one vector empty then you pop again then the function becomes linear to shift the items over.
  + Tricky part will be the indexing that maps across the two vectors correctly. VecOne will be n-1-I. Then the vecTwo will be I – n.