* Homework 2 answer
  + When deck is empty make deck return an invalid card, rank 0 suit spade. Instead of ace of spade. //ugg should have known that....
  + Modify human player draw. Only display cards that are not equal to 0. eliminating the option for the player to play that card. When they select the card check to make sure their answer is valid and make sure the card isnt 0, otherwise their answer is not valid.
  + Change player into the same format as human player. Making the 0 not a valid option. While(mycards[removedcard].getrank() == 0) {removedcard = randomizer(3);}
  + In player make empty to make sure the hand is empty by testing each card in hand is not equal to empty.
  + Should have been able to get that correct -.-
* Merge sorts
  + All the generic functions are from <algorithm>.
  + It is a pretty simplistic function logically
  + T(n) tme for m\_sort for n elements to sort.
    - T(n) = T(n/2) + T(n/2) + cn + cn (simplify the 2 cns to just cn)
    - = 2T(n/2) +cn
    - T(n/2) = 2(2T(n/4) + c(n/2) + cn
      * 4T(n/4) + cn + cn
      * 4T(n/4) + 2cn
    - T(n/4) = 2T(n/4/2) + cn/4
      * 4(2T(n/8) + cn/4) + 2cn
      * 8T(n/8) + cn + 2cn
      * 8T(n/8) + 3cn
      * If n = 8 then T(n) = nT(1) + logncn
        + O(nlogn)
* Lab 7 Deque
  + Deque – Very similar to a vector but you have both push\_back and pop\_back. Push\_front and pop\_front. As well as having an indexing operator. '[]'.
  + Deques are two vectors back to back to each other. Allowing the pops and pushs to work as intended.
  + The price of the deques is if one of the vectors become empty and you try to pop back or pop front off the next element its going to take O(n) in order to shift the vector properly
  + Class Deque{
    - Vector<T> vecOne;
    - Vector<T> vecTwo;
    - Public:
      * ---
      * ---
      * --
    - }
    - Most of the functions will be covered through the book, or in lab. There is only one function that will be required to be written.
* Study Guide exercises:
  + Chapter 7 String - Anigrams:
    - Write a function that takes two words and determines if the functions are anigrams. IE Rescue and secure.
    - Hint use the generic algorithm sort in order to sort the strings. Page 136
    - Sort(iterator start, iterator stop);
    - Bool anigram(string s, string t)
    - {
      * Sort(s.begin(), s.end()); //Does string have a begin and end?
      * Sort(t.begin(), t.end());
      * Return t == s;
    - }
  + Chapter 7 – question 9
    - Average number of words in a sentence through a whole file
    - Int sentenceLength;
    - Int numSent = 0;
    - Int filelength = 0;
    - Cin >> temp;
    - While (!cin.eof()){
      * SentenceLength = 0;
      * While(lastchar != '.')
        + SentenceLength++;
        + Cin >> temp;
      * NumSent++;
      * Filelength += sentenceLength;
      * Cin >> temp;
      * }
      * This may not be completely finished
  + Vectors
    - Problem 16 – page 183
    - Write the insert method for vector
    - Vector<int> v;
    - Vector<int>::iterator itr;
    - v.insert(itr, 5);
    - Think with reserver, resize or push back..
    - This is a weird one...
  + Lists
    - Page 215 – problem 4
    - Write a method for List so it would return the int item in the list.
    - List<int> l;
    - l.intvalue(3); or l[3];
    - Start at the beginning with the iterator. Then have a counter that is less then the given number. Then when the counter is equal to the given number return that value.