Simon Chen
sc4900
Homework 4A

1) O(n)

2)O(n)

3) The erase will invalidate all iterators that come after that iterator in place, inclusive. The iterators that are before the erase iterator are still valid.
e.g.

eraseItr
valid : [begin,eraseItr)
invalid : [eraseItr, end)

4)No, it will contain a garbage value, whatever was in memory there before. In some compilers, they will zero out the vector when it is initialized. The iterator does not contain the value that is stored in c[0]. It contains the address of the object it was before. If it changes, the iterator's value won't change, unless it's changed through the iterator.

5)It will convert the 110 to vector(110), making it equal to a vector with a size of 110 + the SPARE_CAPACITY, which is 2. The cout will return 112.

6)

a)copy(**A.begin(),A.begin()+6,D.begin()**);

b)cout << count(**B.begin(),B.end()**,1);

c)cout << count_if(**B.begin( ), B.end()**, bind1st(not_equal_to<int>( ), 1));

```
d)vector<int>::iterator vecItr;
  vecItr = find(A.begin(), A.end(), 5);
       // returns an iterator to 5 in vector A
  if (vecItr != A.end( ))
       cout << *vecItr;
       // prints out the value pointed to by vecItr
```

```
e)vecItr = find_if(C.begin(), C.end(),bind2nd(greater<int>(), 2));
  // returns an iterator to 3 in C
  if (vecItr != C.end( ))
       cout << *vecItr;
       // prints out the value pointed to by vecItr
```

f)reverse(C.begin(), C.end());

g)sort(B.begin( ), B.end());

h)mismatch(A.begin(), A.end(), C.begin(), equal_to<int>())
//returns a pair of int that tells the mismatched pairs in vector A and vector B. The equal_to

functor is used to compare the two vectors or objects in general.