* Homework 3
  + Main
    - List int v
    - Int temp
    - While cin >> temp
      * v.pushback(temp);
    - Mergesort(v)
    - For (to print list after sort)
  + Mergesort list <T> & s
    - M\_sort(s.begin(), s.end(), s.size())
  + Void m\_sort(itr start, itr stop, int size)
    - If size > I
      * Int first half = size/2
      * Int second half = size – first half
    - Itr center = start;
    - For int I = 0; I < first half; I++
      * Center++;
    - Msort(start, center, first half)
    - Msort(center, stop, second half)
    - Inplace merge(start, center, stop);
  + Void inplacemerge(itr start, itr center, itr stop)
    - List<typename> iterator\_traits<Itr>::value\_type temp;
    - Itr I = low;
    - Itr j = mid;
    - While(I !=mid && j != high)
      * If( \*I > \*j)
        + Temp.push\_back(\*I++)
      * Else
        + Temp.pushback(\*j++)
    - While( I != mid)
      * Temp.pushback(\*I++);
    - While (j != high)
      * Temp.pushback((\*j++);
    - For j = temp.begin(), I = low,; I != temp.end(); j++, I++)
      * \*I = \*j;
* Homework 4
  + To move half the elemts from vecTwo to vecOne you have to use reserve
    - VecOne.reserve(vecTwo.size/2)
    - Copy from vecTwo to vecOne
    - Shift in vecTwo.
    - Then vecOne.pop\_back
* Maps
  + #include <map>
  + #include <string>
  + Void readItems(map<string,int> & m)
    - String word;
    - Int value;
    - While(cin>>word>>value)
      * M[word] += val;
    - // note when executing file it would be a.out < hardware.doc
  + Main()
    - Map<string, int> tbl;
    - ReadItems(tbl);
    - Int total = 0;
    - Map<string, int>::const\_iterator p;
    - For(p = tbl.begin(); p != tbl.end(); p++)
      * Total += p->second; //note maps have first(key) and second(value);
      * Cout << p -> first << " " << p-> second << endl;
    - Cout << total << endl;
* Maps lab
  + Erase
    - Void erase(const key & k) { parent::erase(Pair<key,value>(k,value()); }
  + Iterator find(const key & k) { return parent::find(Pair<key, value>(k,value())); }
  + Value & operator[](const key & k)
    - Iterator it = parent::insert(Pair<key,value>(k,value()) );
    - Return (\*it).second;