```
//
//  main.cpp
//  PointerPractice
//
//  Created by Nathan Sturtevant on 1/23/14.
//  Copyright (c) 2014 Nathan Sturtevant. All rights reserved.
//

#include <iostream>

const int arraySize = 100;
using std::cout;
using std::endl;

void sort(int *front, int numItems);
void merge(int *result, int *list1, int list1size, int *list2, int list2size);

int main(int argc, const char * argv[])
{
    int numbers[arraySize];

    for (int x = 0; x < arraySize; x++)
    {
        numbers[x] = random()%(arraySize*10);
    }

    sort(&numbers[0], arraySize);

    for (int x = 0; x < arraySize; x++)
    {
        cout << numbers[x] << endl;
    }

    return 0;
}

void sort(int *front, int numItems)
{
    if (numItems <= 1)
        return;
    int *firstHalf = front;
    int *secondHalf = &front[numItems/2];
    int firstSize = numItems/2;
    int secondSize = numItems-numItems/2;

    sort(firstHalf, firstSize);
    sort(secondHalf, secondSize);

    int *tmp = new int[numItems];
    merge(tmp, firstHalf, firstSize, secondHalf, secondSize);
    for (int x = 0; x < numItems; x++)
        front[x] = tmp[x];//*(tmp+x);
    delete [] tmp;
}
```



```
void merge(int *result, int *list1, int list1size, int *list2, int list2size)
{
    if (list1size == 0 && list2size == 0)
    {
        return;
    }
    else if (list1size == 0)
    {
        *result = *list2;
        ++result;
        ++list2;
        list2size--;
        merge(result, list1, list1size, list2, list2size);
    }
    else if (list2size == 0)
    {
        merge(result, list2, list2size, list1, list1size);
    }
    else if (*list1 >= list2[0])
    {
        *result = *list2;
        ++result;
        ++list2;
        list2size--;
        merge(result, list1, list1size, list2, list2size);
    }
    else {
        merge(result, list2, list2size, list1, list1size);
    }
}
```