1. The idea about doing sorting by merge sort .

   First, we divided an array into many subarray, note that our main is to divide the array until all the array become only have one element.

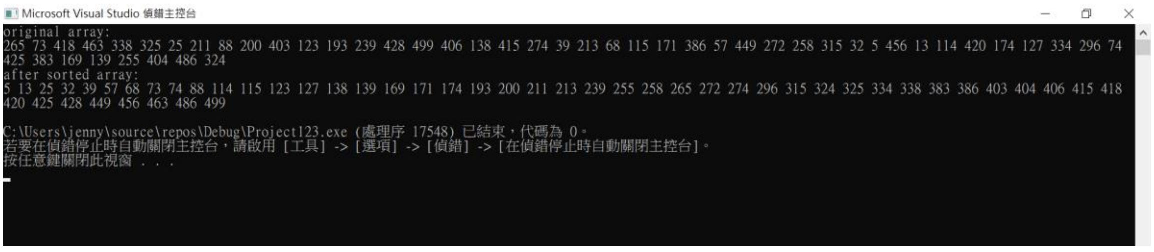
   Next step, we are going to merge the divided array, at the same time, we will sorting the array, the way merged the array is that we choose the first element of the two merge array, compare them, then put the smaller one back to the original array, after the element is put back to the original array, we choose the next element in the same array, repeat this step until all the subarray be merge into one.

2. About divide the array, let an array contain n elements divide into n array each contain only one element need to do (n-1) time., so the complexity of merge are equal to (n-1).

   About merge the array and sorting at the same time, merge n number into one array, we need n steps, and the n steps merge we need to do logn time, so the complexity of merge are equal to nlogn.

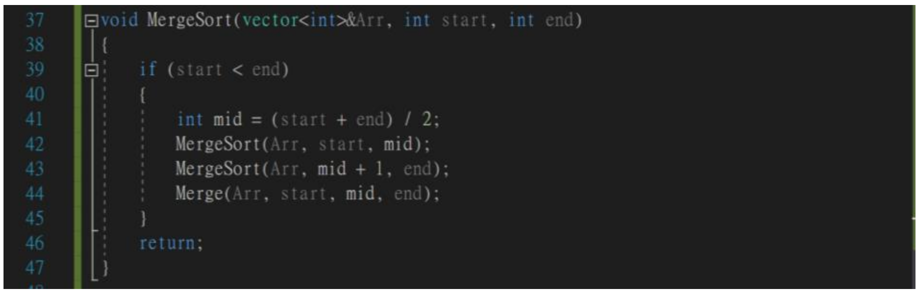
   To sum up, sorting an n elements array by merge sort, plus the steps about divide and merge, we need (n-1)+(nlogn) steps to finish the sorting, so the complexity of merge sort equal to nlogn.

3. 50 個 random number 使用 merge sort 做 sorting 後的結果(before & after)(10%)



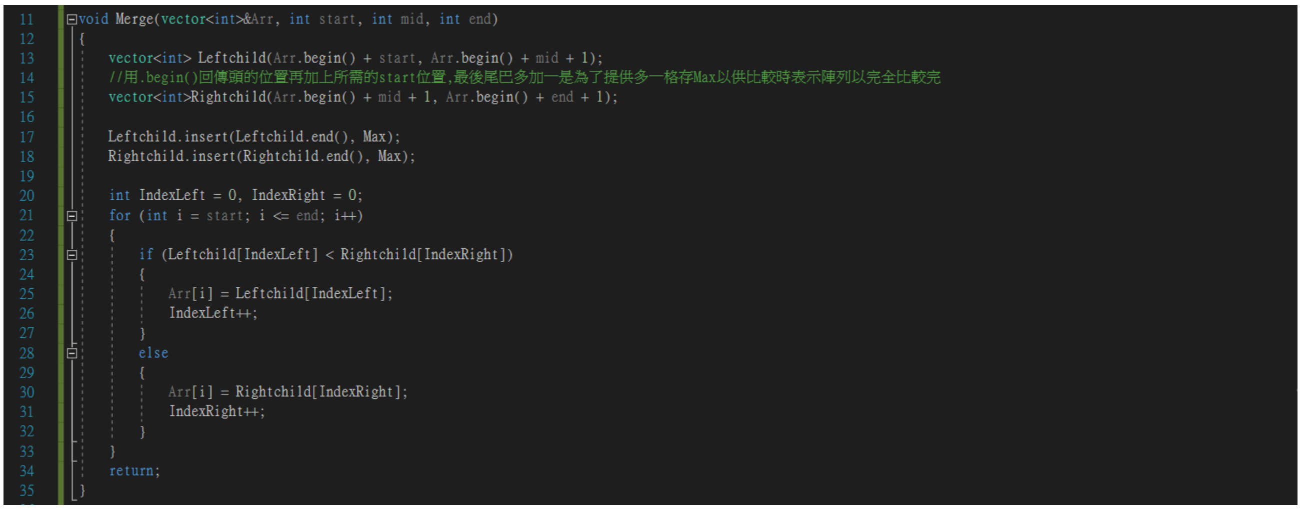
```
original array:
265 73 418 463 338 325 25 211 88 200 403 123 193 239 428 499 406 138 415 274 39 213 68 115 171 386 57 449 272 258 315 32 5 456 13 114 420 174 127 334 296 74
425 383 169 139 255 404 486 324
after sorted array:
5 13 25 32 39 57 68 73 74 88 114 115 123 127 138 139 169 171 174 193 200 211 213 239 255 258 265 272 274 296 315 324 325 334 338 383 386 403 404 406 415 418
420 425 428 449 456 463 486 499

C:\Users\jenny\source\repos\Debug\Project123.exe (處理序 17548) 已結束，代碼為 0。
若要在偵錯停止時自動關閉主控台，請啟用 [工具] -> [選項] -> [偵錯] -> [在偵錯停止時自動關閉主控台]。
按任意鍵關閉此視窗 . . .
```

4. 用文字解釋如何使用程式碼實作 merge sort(實作細節需如同附錄的 pdf 所述)(50%)



```cpp
void MergeSort(vector<int>&Arr, int start, int end)
{
    if (start < end)
    {
        int mid = (start + end) / 2;
        MergeSort(Arr, start, mid);
        MergeSort(Arr, mid + 1, end);
        Merge(Arr, start, mid, end);
    }
    return;
}
```

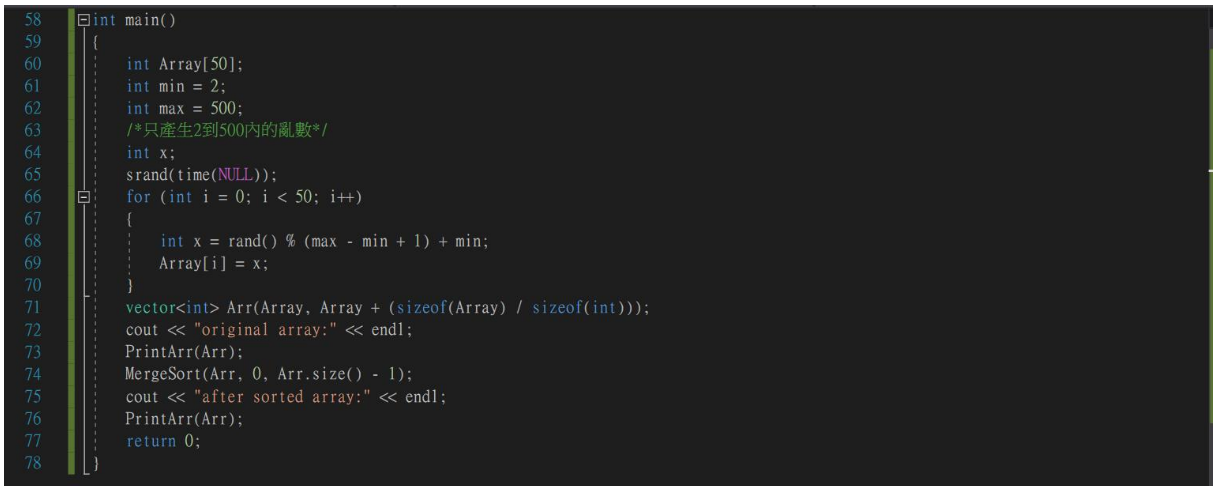
First, create a recursive function called MergeSort, this is the main implement function to do the Merge Sort. In my code , I use the method that divide the array from the mid each time, until each array only contain one element, in other word, Incompatible the condition (start<end).



```cpp
void Merge(vector<int>&Arr, int start, int mid, int end)
{
    vector<int> Leftchild(Arr.begin() + start, Arr.begin() + mid + 1);
    //用.begin()回傳頭的位置再加上所需的start位置,最後尾巴多加一是為了提供多一格存Max以供比較時表示陣列以完全比較完
    vector<int>Rightchild(Arr.begin() + mid + 1, Arr.begin() + end + 1);

    Leftchild.insert(Leftchild.end(), Max);
    Rightchild.insert(Rightchild.end(), Max);

    int IndexLeft = 0, IndexRight = 0;
    for (int i = start; i <= end; i++)
    {
        if (Leftchild[IndexLeft] < Rightchild[IndexRight])
        {
            Arr[i] = Leftchild[IndexLeft];
            IndexLeft++;
        }
        else
        {
            Arr[i] = Rightchild[IndexRight];
            IndexRight++;
        }
    }
    return;
}
```

Second, we do the action that merge each one element array into one array, select the element from the start of both Leftchild and Rightchild, each time compare, than choose bigger one put into the merge array, and change the select element to

the next element of the child.



```
int main()
{
    int Array[50];
    int min = 2;
    int max = 500;
    /*只產生2到500內的亂數*/
    int x;
    srand(time(NULL));
    for (int i = 0; i < 50; i++)
    {
        int x = rand() % (max - min + 1) + min;
        Array[i] = x;
    }
    vector<int> Arr(Array, Array + (sizeof(Array) / sizeof(int)));
    cout << "original array:" << endl;
    PrintArr(Arr);
    MergeSort(Arr, 0, Arr.size() - 1);
    cout << "after sorted array:" << endl;
    PrintArr(Arr);
    return 0;
}
```

The main function, we use random method to create a fifty element array, than call

the MergeSort function to do the array merge sorting.