2019/3/19 XX1072078 hw1

1. 程式概念

- 下方為file tree

```
| - - hw1
|     | - -hw1.cpp
|     | - -input.txt
|     | - -output.txt
|     | - -XX1072078_hw1_v2.pdf
```

- 使用方式

```
$ cd
$ git clone https://github.com/tony92151/algorithm_homework.git
$ cd ~/algorithm_homework/hw1
$ g++ hw1.cpp && ./a.out
```

- Input & output 格式（以空格隔開）

  10 68 12 19 53 0 11 4 67 79 32

  其中第一個數字定義這個資料的長度

- 在main中
  A. 定義輸入及輸出文件
  B. 讀取第一個數值length，再定義一個int array，長度為length
  C. 顯示長度及原始array
  D. 進行MergeSort
  E. 顯示新array，並儲存進output.txt （以空格隔開）

- 在MergeSort 及 Merge中，架構與講義相同

```
MERGE-SORT(A, p, r)
1   if p < r
2         q = ⌊(p+r)/2⌋
3         MERGE-SORT(A, p, q)
4         MERGE-SORT(A, q+1, r)
5         MERGE(A, p, q, r)
```

```
MERGE(A, p, q, r)
1   n1 = q – p + 1
2   n2 = r – q
3   create arrays L[1 .. n1+1] and R[1 .. n2+1]
4   for i = 1 to n1
5       L[i] = A[p + i – 1]
6   for j = 1 to n2
7       R[j] = A[q + j]
8   L[n1 + 1] = ∞
9   R[n2 + 1] = ∞
10  i = 1
11  j = 1
12  for k = p to r
13      if L[i] ≤ R[j]
14          then A[k] = L[i]
15               i = i + 1
16          else A[k] = R[j]
17               j = j + 1
```

```
~/Documents/github_project/algorithm_homework/hw1  master ● ?  g++ hw1.cp
p && ./a.out
Length:10
old array : 68  12  19  53  0  11  4  67  79  32
new array : 0  0  4  11  12  19  32  53  67  68
store new array in optput.txt
Done
~/Documents/github_project/algorithm_homework/hw1  master ● ?
```

2. Code

```
1.  /**
2.   *  Name : hw1
3.   *
4.   *  Author : Tony Guo
5.   *
6.   *  Country : Taiwan
7.   *
8.   *  Date : 18 Mar, 2019
9.   *
10.  *  github : https://github.com/tony92151/algorithm_homework
11.  */
12.
13.
14. #include <stdio.h>
15. #include <iostream>
16. #include <fstream>
17. #include <string>
18. using namespace std;
19.
20.
21. void MergeSort(int A[],int p,int r);
22. void Merge(int A[], int p, int q, int r);
23.
24.
25. int main(){
26.
27.     ifstream input( "input.txt" );
28.     ofstream output( "output.txt" );
29.
30.     int leagth;
31.     int array;
32.
33.     if (input.is_open()){
34.         bool first = true;
35.
36.         //read first line to get length
37.         input >> leagth;
```

```
38.         cout<<"Length:" << leagth <<"\n";
39.
40.         //define array length
41.         int *array = new int [leagth];
42.
43.         //read each number in array
44.         int count = 0;
45.         while (count < leagth && input >> array[count]) count++;
46.
47.         //display array
48.         cout<<"old array : ";
49.         for(int i = 0; i < leagth; i++) cout<<array[i]<<"  ";
50.         cout<<"\n";
51.
52.         //sort
53.         MergeSort(array, 0, 10);
54.
55.         //display new array
56.         cout<<"new array : ";
57.         for(int i = 0; i < leagth; i++) cout<<array[i]<<"  ";
58.         cout<<"\n";
59.
60.         //store new array in optput.txt
61.         cout<<"store new array in optput.txt\n";
62.         count = 0;
63.         output <<  leagth <<"\n";
64.         while (count < leagth && output<<array[count]<<"\n") count+
    +;
65.         cout<<"Done\n";
66.
67.
68.     }
69.
70. }
71.
72. void MergeSort(int A[], int p, int r) {
73.     if (p < r) {
74.         int h = (p+r)/2;//half
75.         MergeSort(A, p, h);
76.         MergeSort(A, h+1, r);
77.         Merge(A, p, r, h);
78.     }
79. }
80.
81. void Merge(int A[], int p, int q, int r) {
82.     int n1 = r - p + 1;
83.     int n2 = q - r;
84.
85.     //new array
86.     int *L = new int[n1];
87.     int *R = new int[n2];
88.
```

```
89.     for (int i = 0; i < n1; i++) {
90.         L[i] = A[p+i];
91.     }
92.     for (int i = 0; i < n2; i++) {
93.         R[i] = A[i+r+1];
94.     }
95. 
96.     L[n1] = 2147483647;//INF
97.     R[n2] = 2147483647;//INF
98. 
99.     int i = 0, j = 0;
100.    for (int k = p; k <= q; k++) {
101.        if (L[i] <= R[j]) {
102.            A[k] = L[i];
103.            i++;
104.        }
105.        else {
106.            A[k] = R[j];
107.            j++;
108.        }
109.    }
110.}
111.
```