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

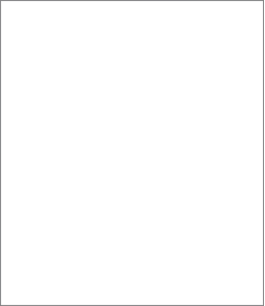
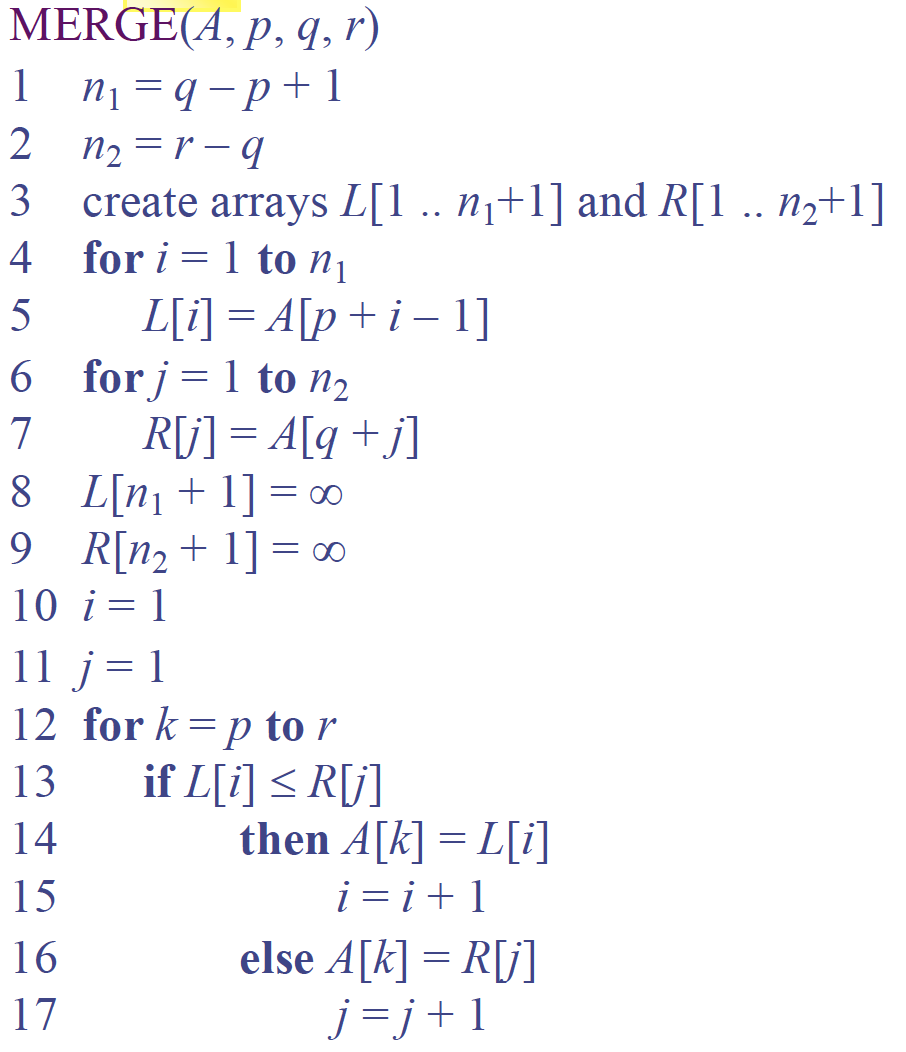
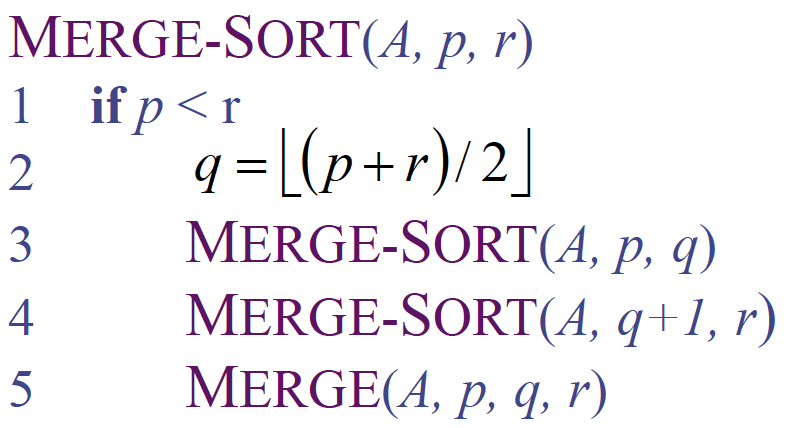
$ g++ hw1.cpp && ./a.out

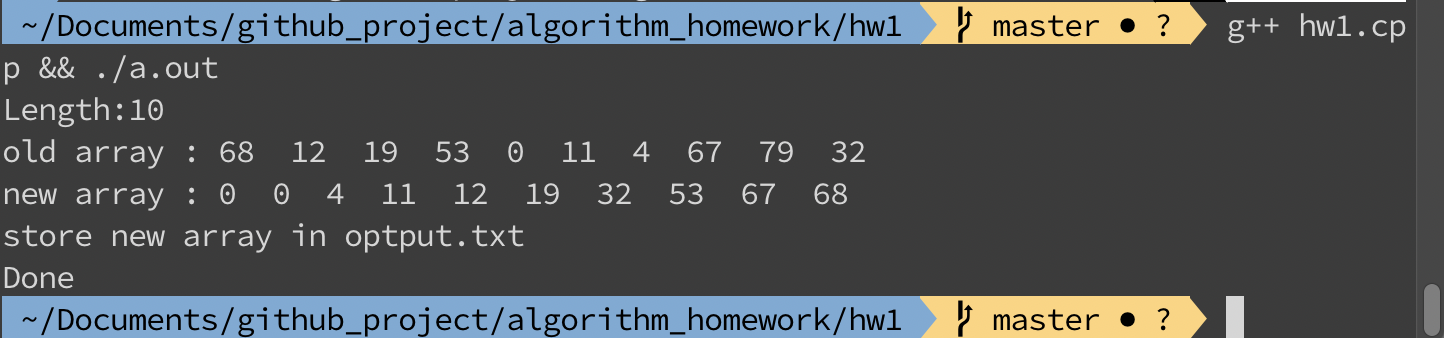
* Input & output 格式（以空格隔開）

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
  1. 定義輸入及輸出文件
  2. 讀取第一個數值length，再定義一個int array，長度為length
  3. 顯示長度及原始array
  4. 進行MergeSort
  5. 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





1. Code
2. /\*\*
3. \*  Name : hw1
4. \*
5. \*  Author : Tony Guo
6. \*
7. \*  Country : Taiwan
8. \*
9. \*  Date : 18 Mar, 2019
10. \*
11. \*  github : https://github.com/tony92151/algorithm\_homework
12. \*/

15. #include <stdio.h>
16. #include <iostream>
17. #include <fstream>
18. #include <string>
19. **using** **namespace** std;

22. **void** MergeSort(**int** A[],**int** p,**int** r);
23. **void** Merge(**int** A[], **int** p, **int** q, **int** r);

26. **int** main(){
28. ifstream input( "input.txt" );
29. ofstream output( "output.txt" );
31. **int** leagth;
32. **int** array;
34. **if** (input.is\_open()){
35. **bool** first = **true**;
37. //read first line to get length
38. input >> leagth;
39. cout<<"Length:" << leagth <<"\n";
41. //define array length
42. **int** \*array = **new** **int** [leagth];
44. //read each number in array
45. **int** count = 0;
46. **while** (count < leagth && input >> array[count]) count++;
48. //display array
49. cout<<"old array : ";
50. **for**(**int** i = 0; i < leagth; i++) cout<<array[i]<<"  ";
51. cout<<"\n";
53. //sort
54. MergeSort(array, 0, 10);
56. //display new array
57. cout<<"new array : ";
58. **for**(**int** i = 0; i < leagth; i++) cout<<array[i]<<"  ";
59. cout<<"\n";
61. //store new array in optput.txt
62. cout<<"store new array in optput.txt\n";
63. count = 0;
64. output << leagth <<"\n";
65. **while** (count < leagth && output<<array[count]<<“\n") count++;
66. cout<<"Done\n";

69. }
71. }
73. **void** MergeSort(**int** A[], **int** p, **int** r) {
74. **if** (p < r) {
75. **int** h = (p+r)/2;//half
76. MergeSort(A, p, h);
77. MergeSort(A, h+1, r);
78. Merge(A, p, r, h);
79. }
80. }
82. **void** Merge(**int** A[], **int** p, **int** q, **int** r) {
83. **int** n1 = r - p + 1;
84. **int** n2 = q - r;
86. //new array
87. **int** \*L = **new** **int**[n1];
88. **int** \*R = **new** **int**[n2];
90. **for** (**int** i = 0; i < n1; i++) {
91. L[i] = A[p+i];
92. }
93. **for** (**int** i = 0; i < n2; i++) {
94. R[i] = A[i+r+1];
95. }
97. L[n1] = 2147483647;//INF
98. R[n2] = 2147483647;//INF
100. **int** i = 0, j = 0;
101. **for** (**int** k = p; k <= q; k++) {
102. **if** (L[i] <= R[j]) {
103. A[k] = L[i];
104. i++;
105. }
106. **else** {
107. A[k] = R[j];
108. j++;
109. }
110. }
111. }