組合語言報告

資工二甲 康智詠

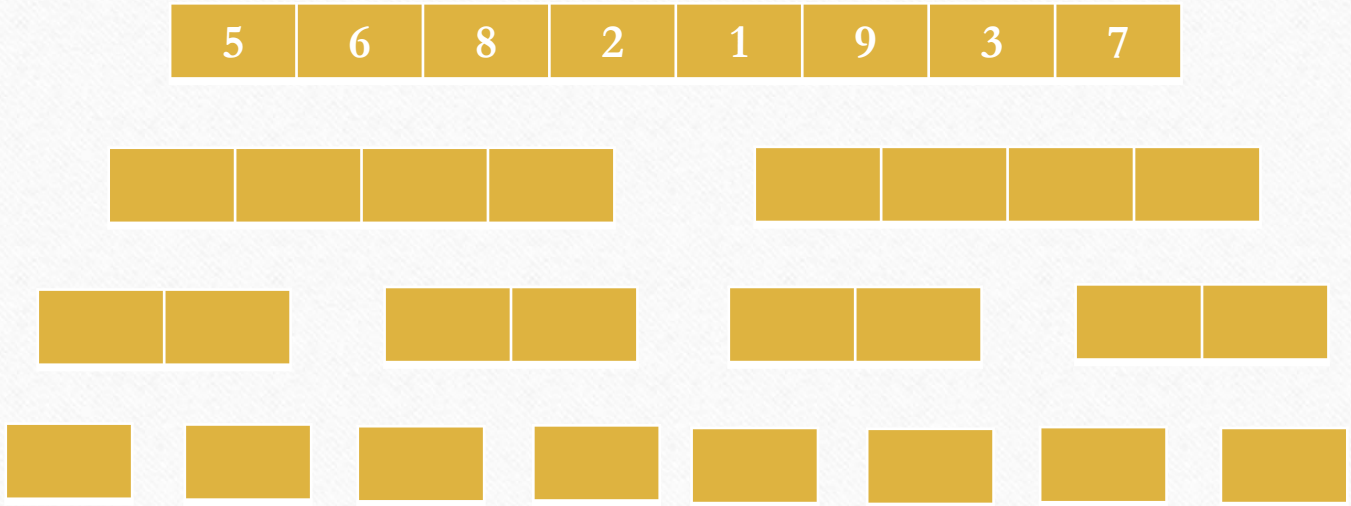
資工二甲 林子傑

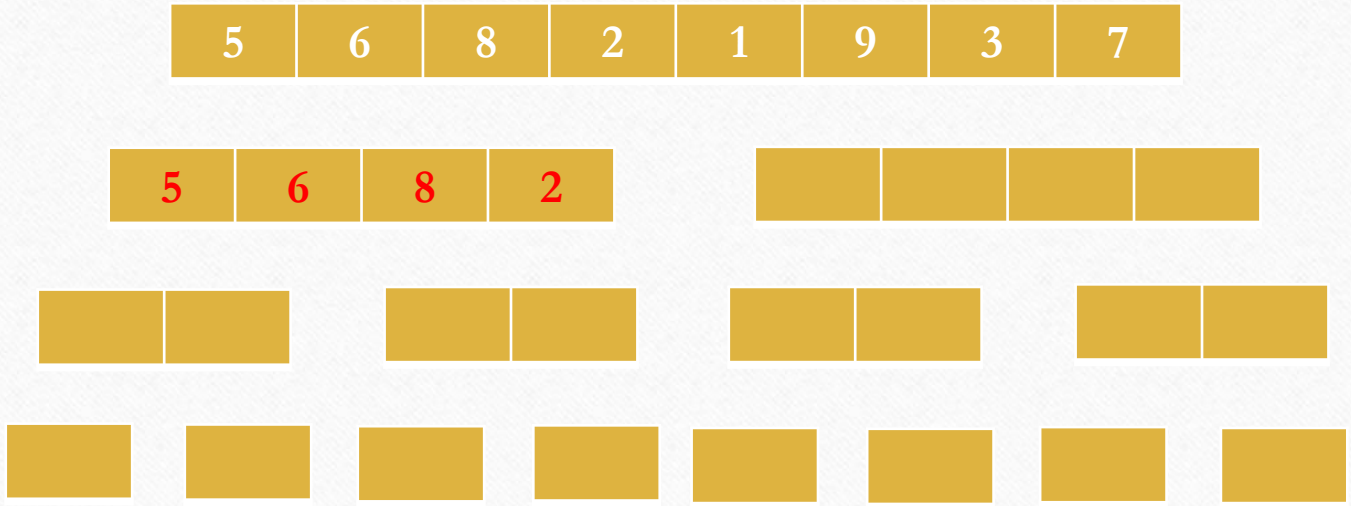
主題

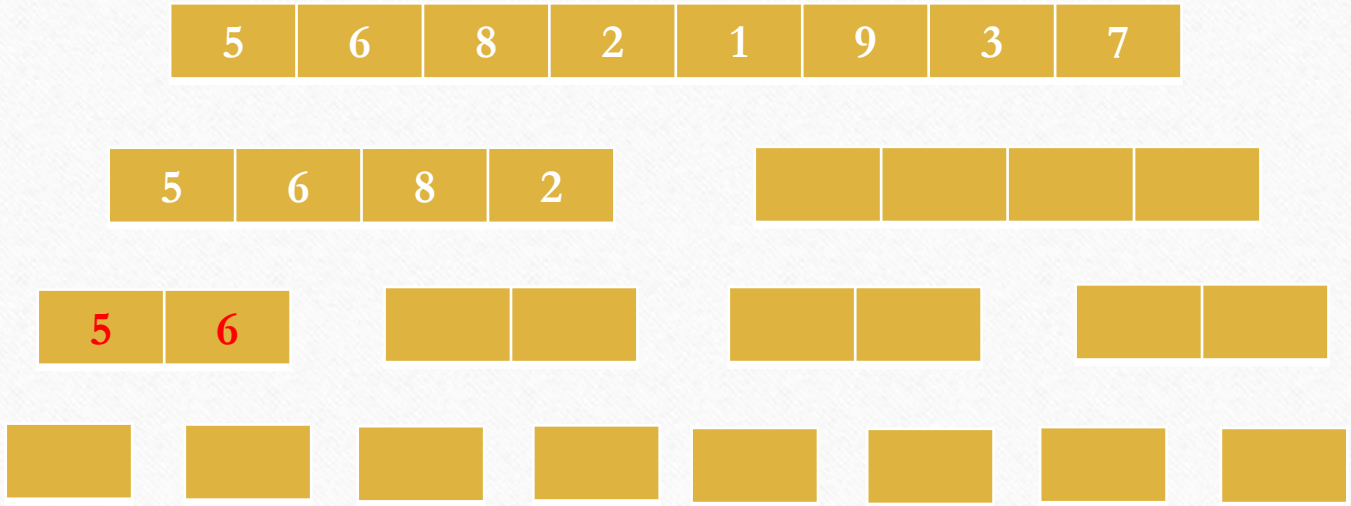
- 比較自己寫和Visual Studio反組譯C/C++語言所產生的組合語言寫的合併排序法(Merge Sort)

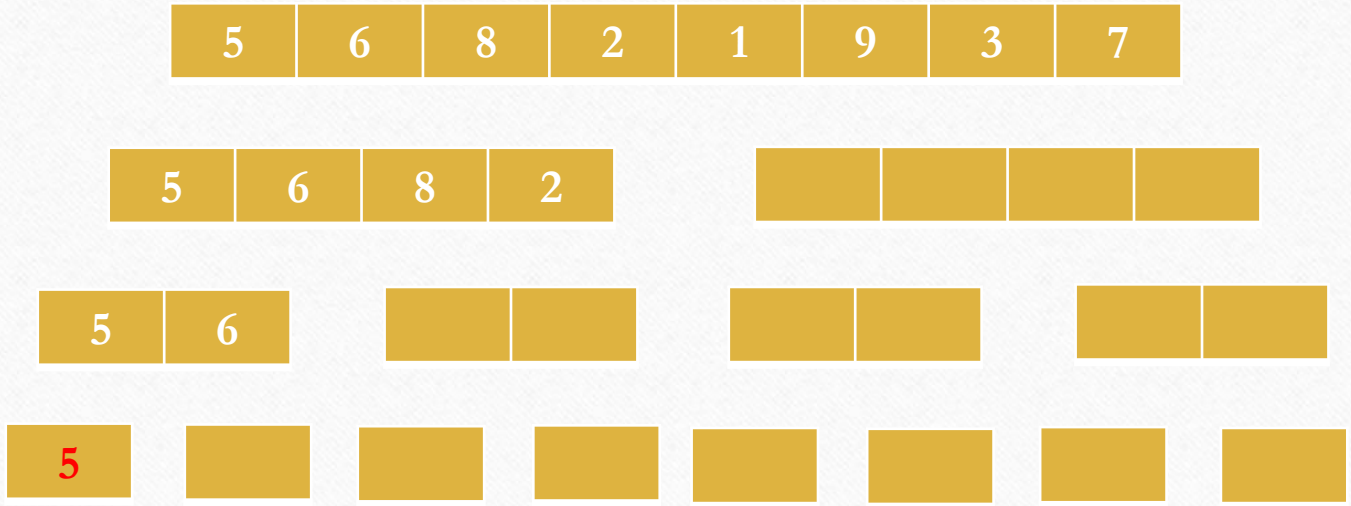
Merge Sort

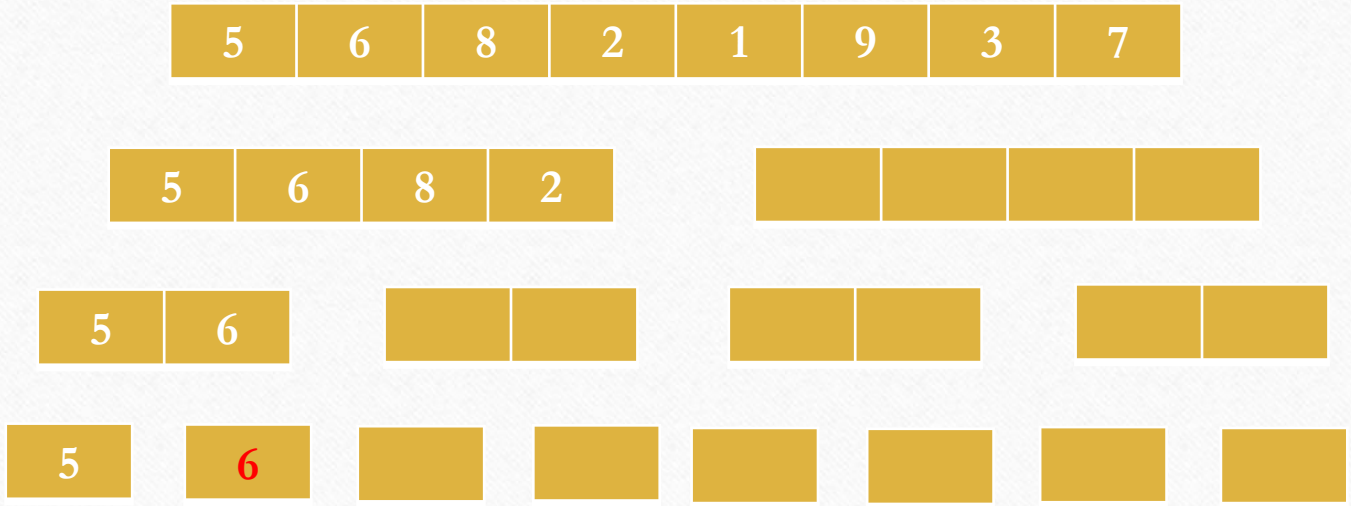
- 用遞迴實作
- 把序列分割直到長度為1
- 向上合併

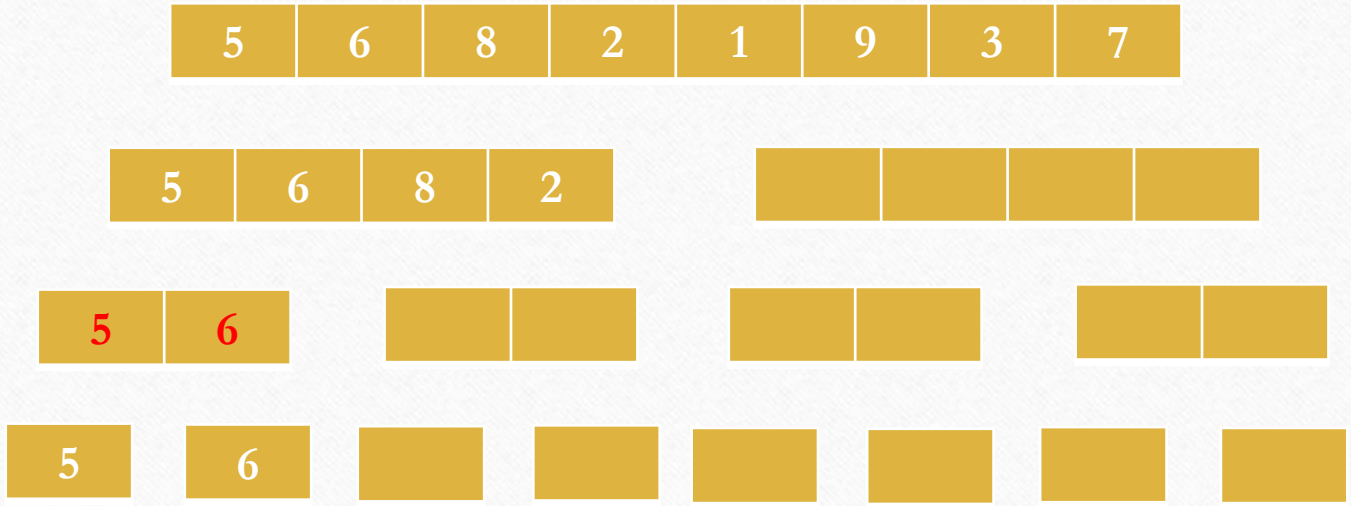

5 6 8 2 1 9 3 7

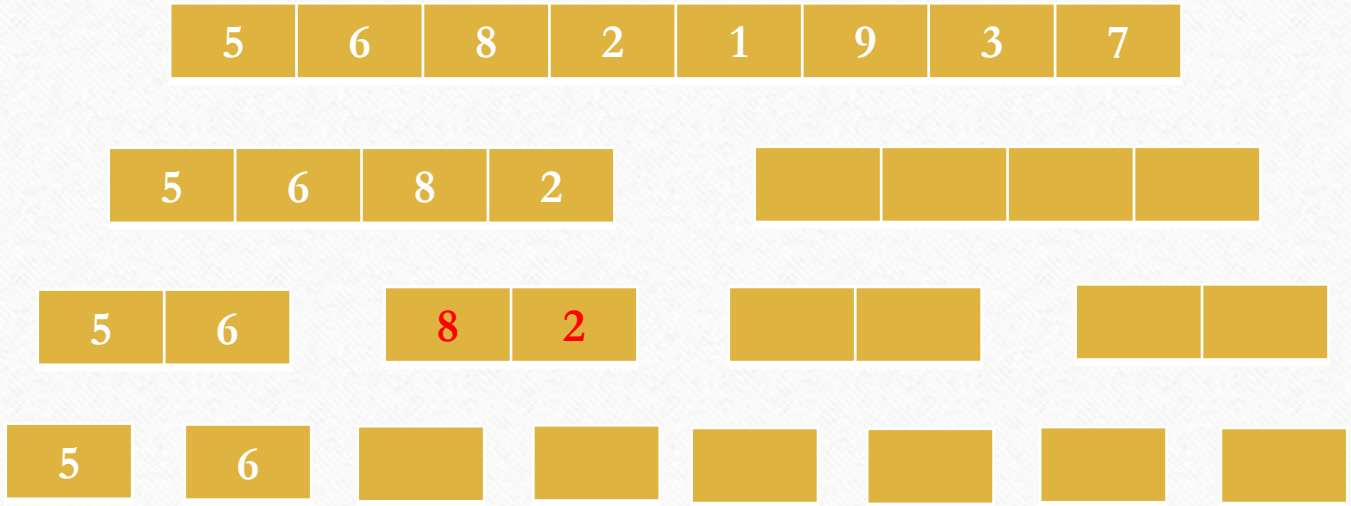

5 6 8 2 1 9 3 7
5 6 8 2

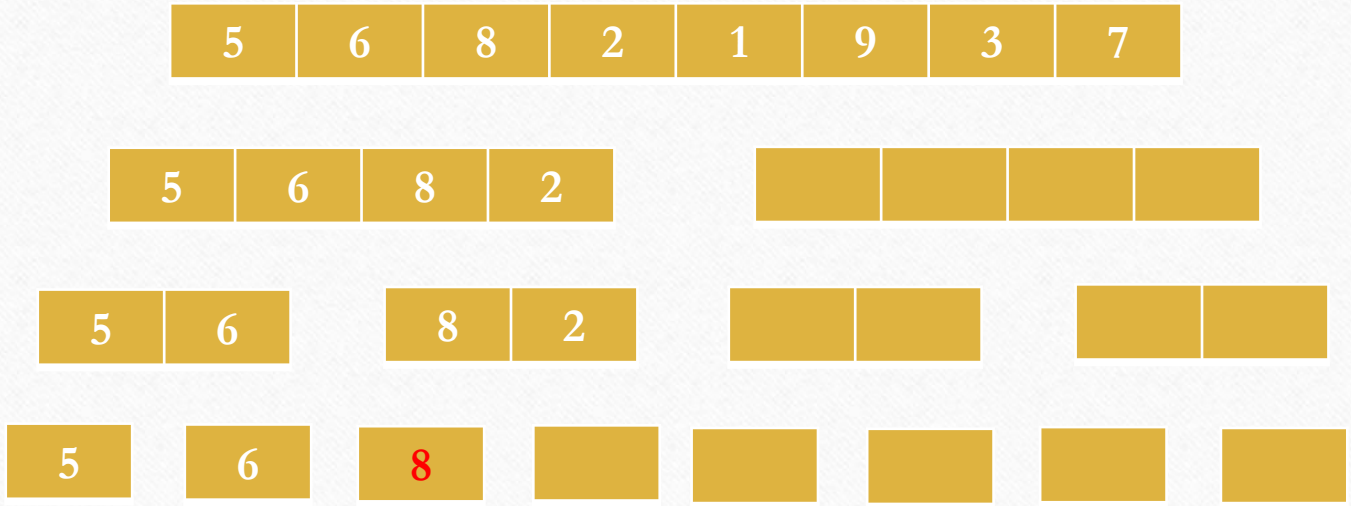

5 6 8 2 1 9 3 7
5 6 8 2
5 6

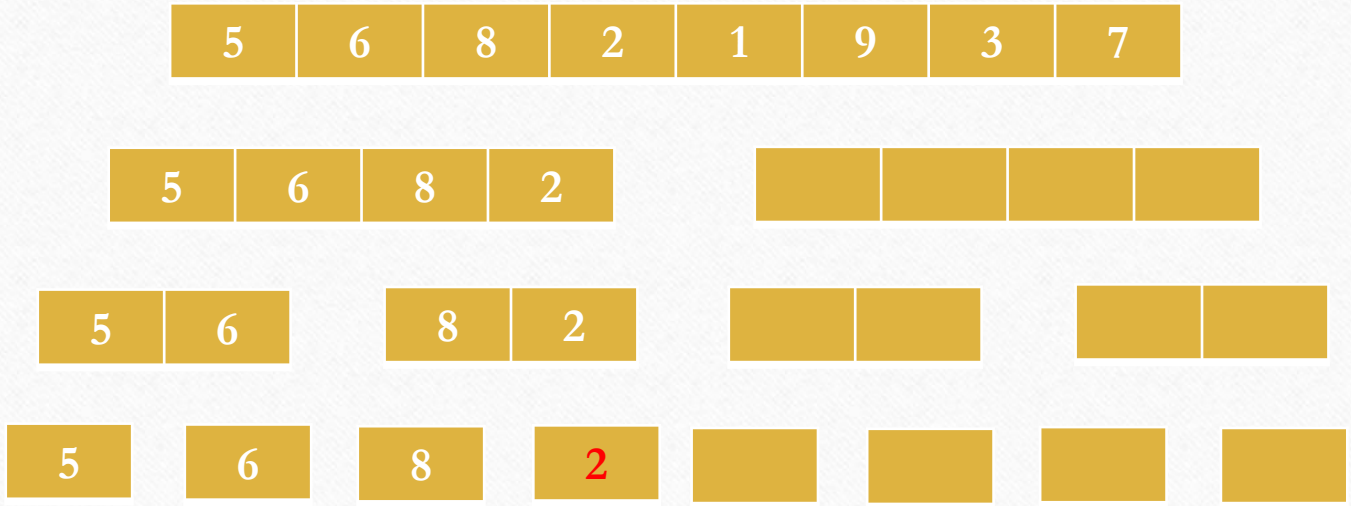

5 6 8 2 1 9 3 7
5 6 8 2
5 6
5

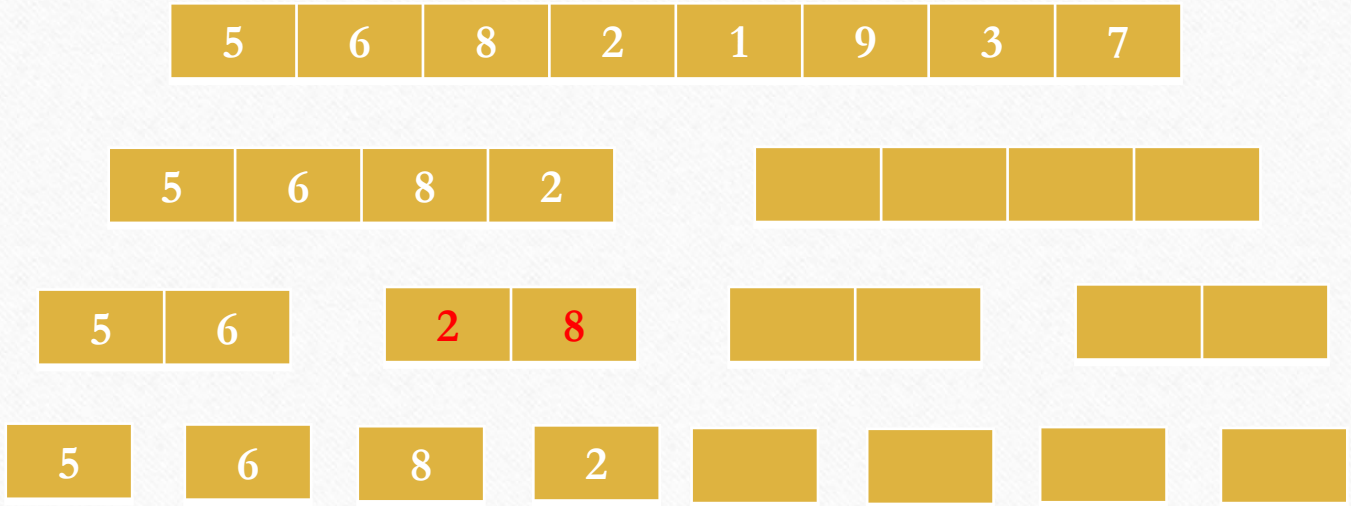

5 6 8 2 1 9 3 7
5 6 8 2
5 6
5
6

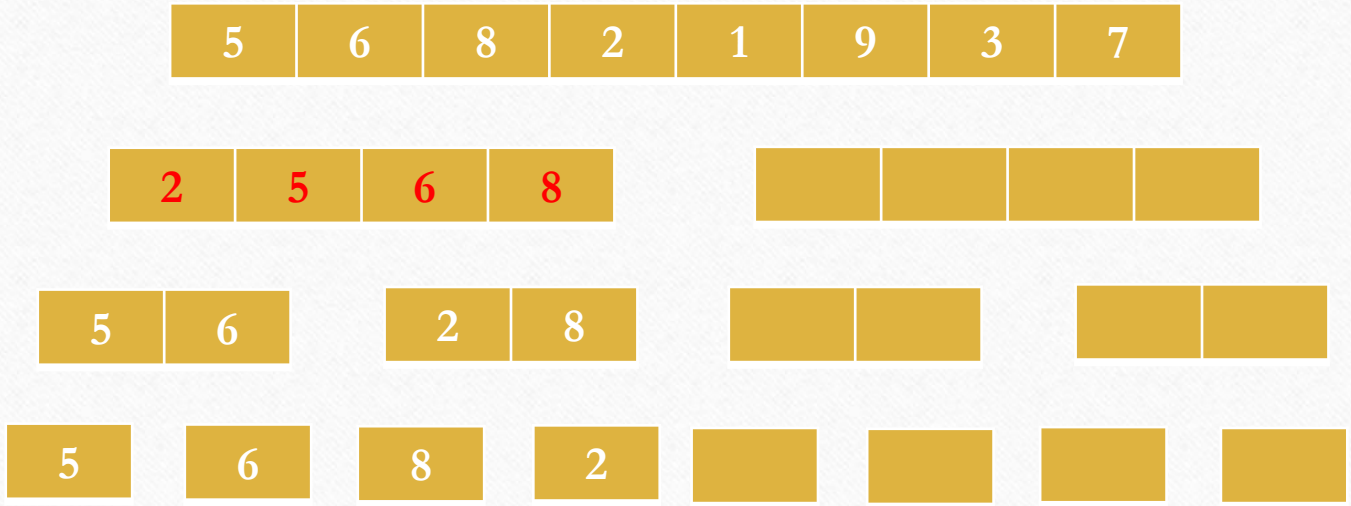

5 6 8 2 1 9 3 7
5 6 8 2
5 6
5 6

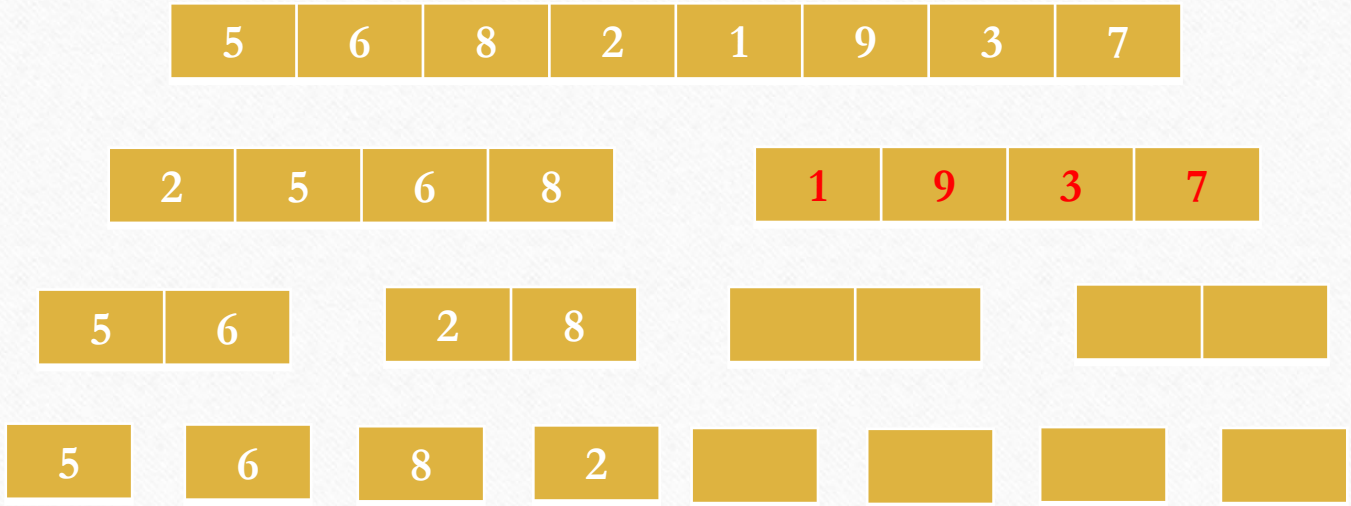

5 6 8 2 1 9 3 7
5 6 8 2
5 6 8 2
5 6

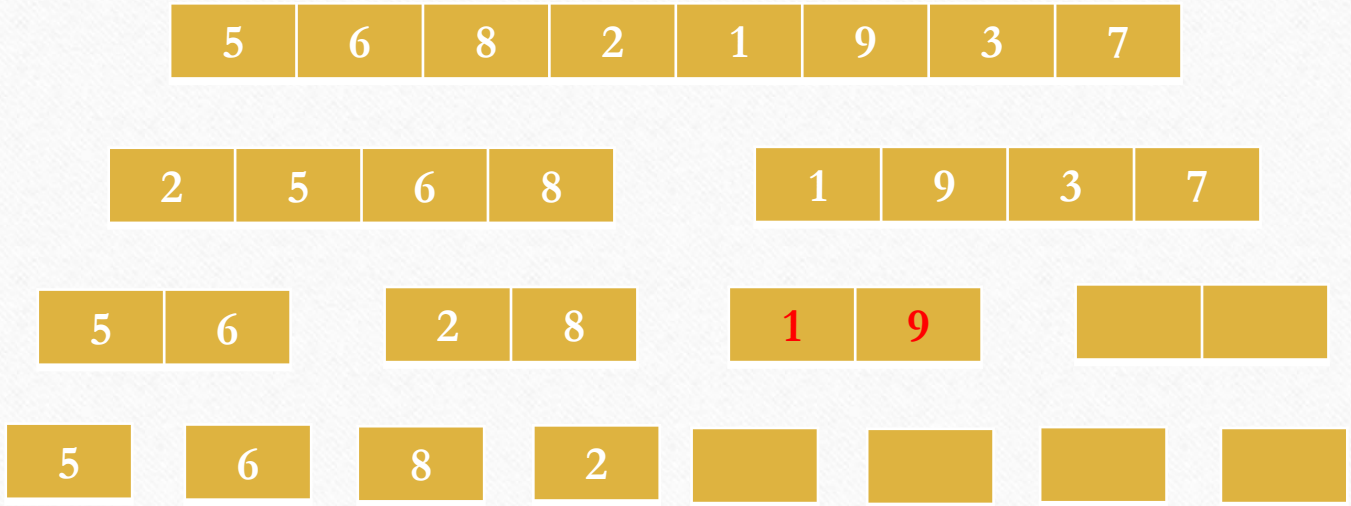

5 6 8 2 1 9 3 7
5 6 8 2
5 6
8 2
5
6
8

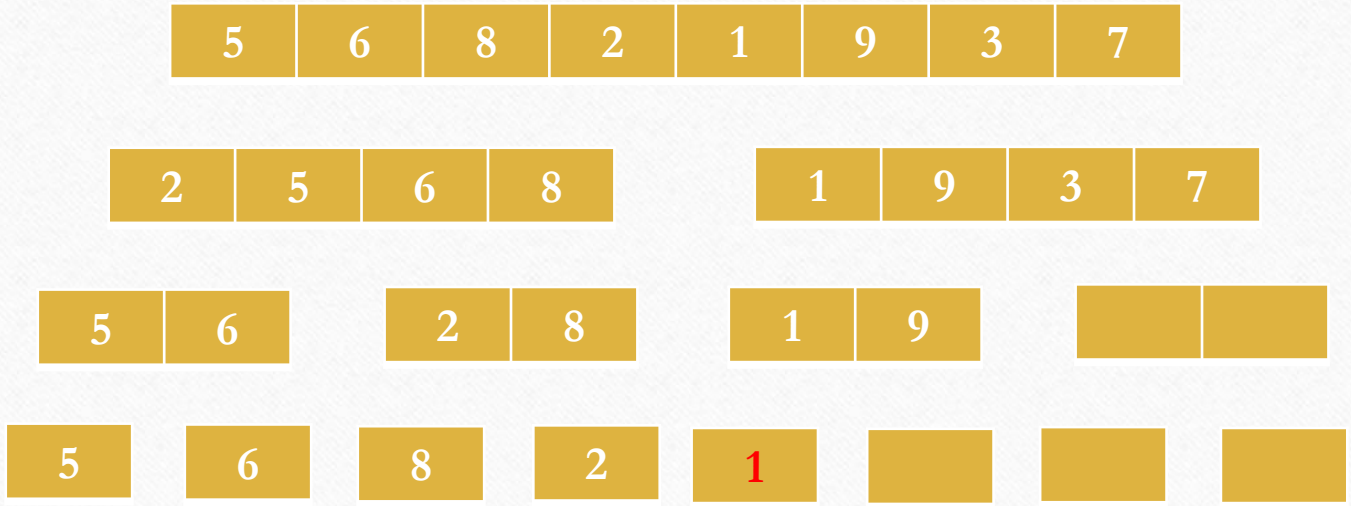

5 6 8 2 1 9 3 7
5 6 8 2
5 6 8 2
5 6 8 2

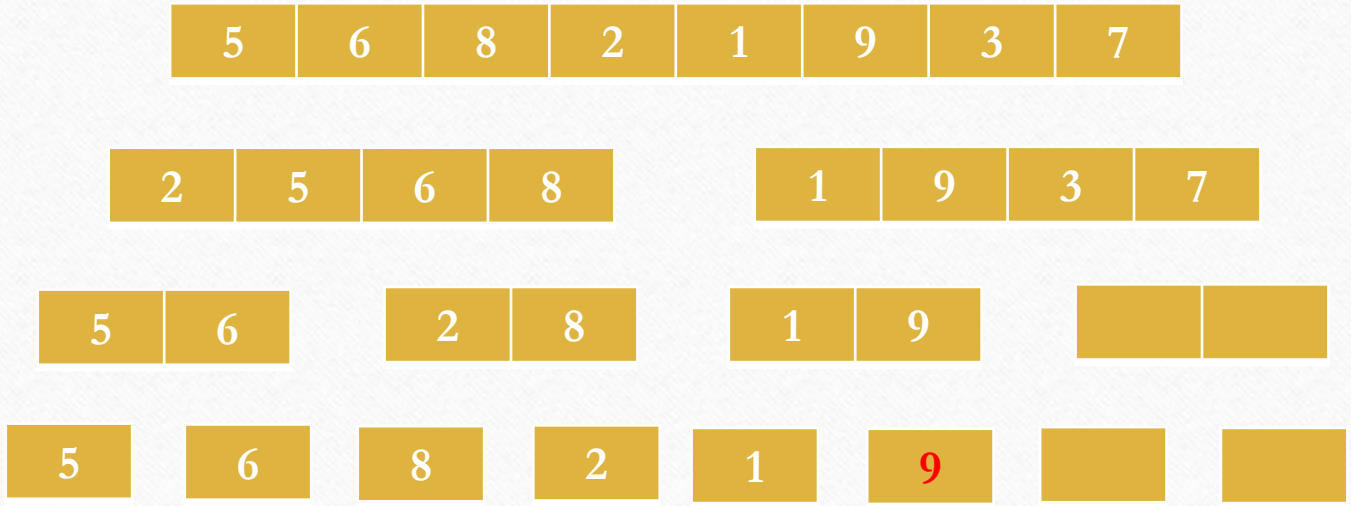

5 6 8 2 1 9 3 7
5 6 8 2
5 6 2 8
5 6 8 2

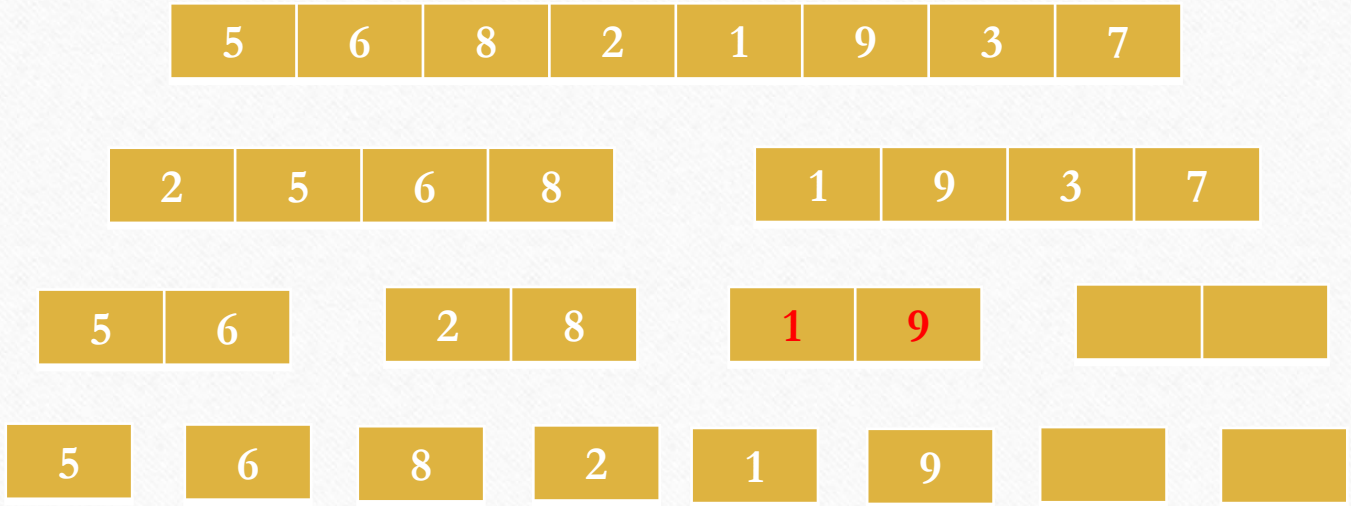

5 6 8 2 1 9 3 7
2 5 6 8
5 6
2 8
5
6
8
2

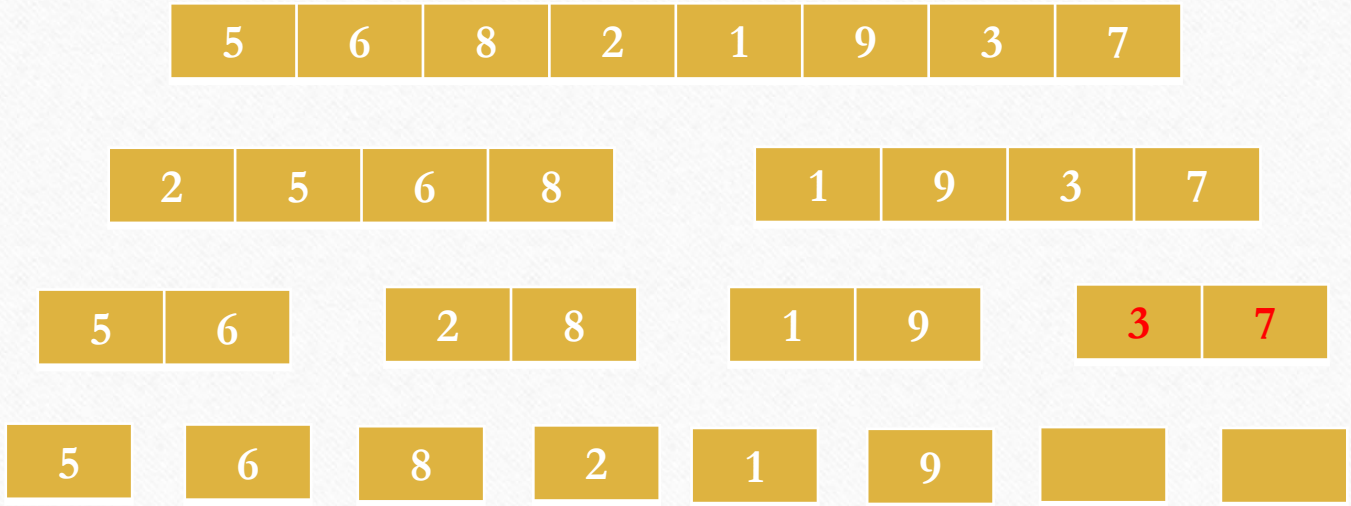

5 6 8 2 1 9 3 7
2 5 6 8
1 9 3 7
5 6
2 8

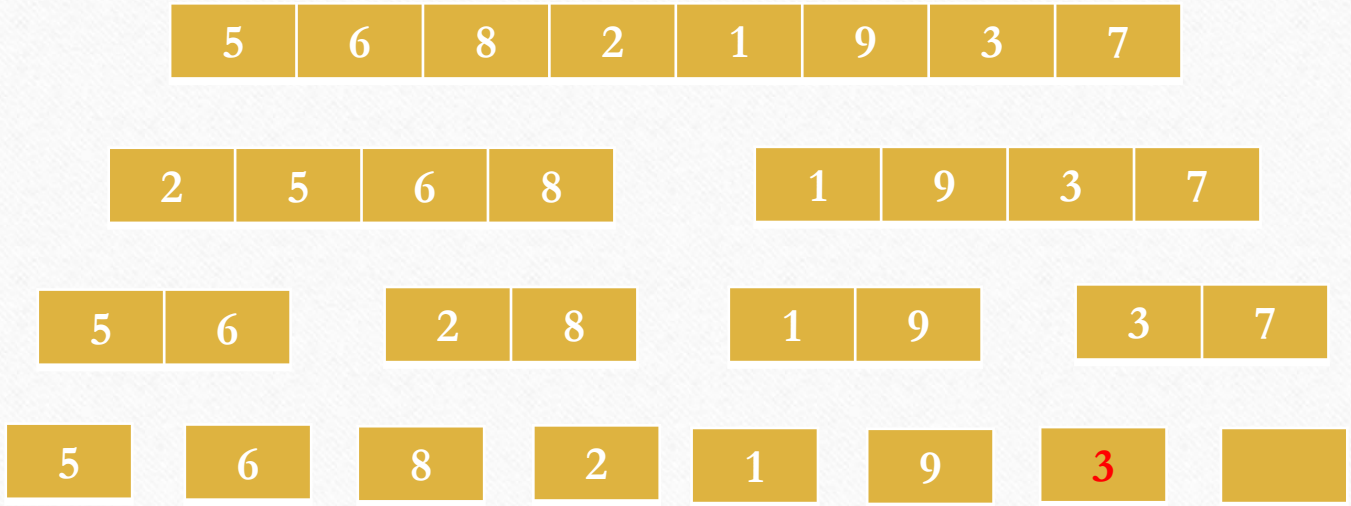

5 6 8 2 1 9 3 7
2 5 6 8
1 9 3 7
5 6
2 8
1 9
5
6
8
2

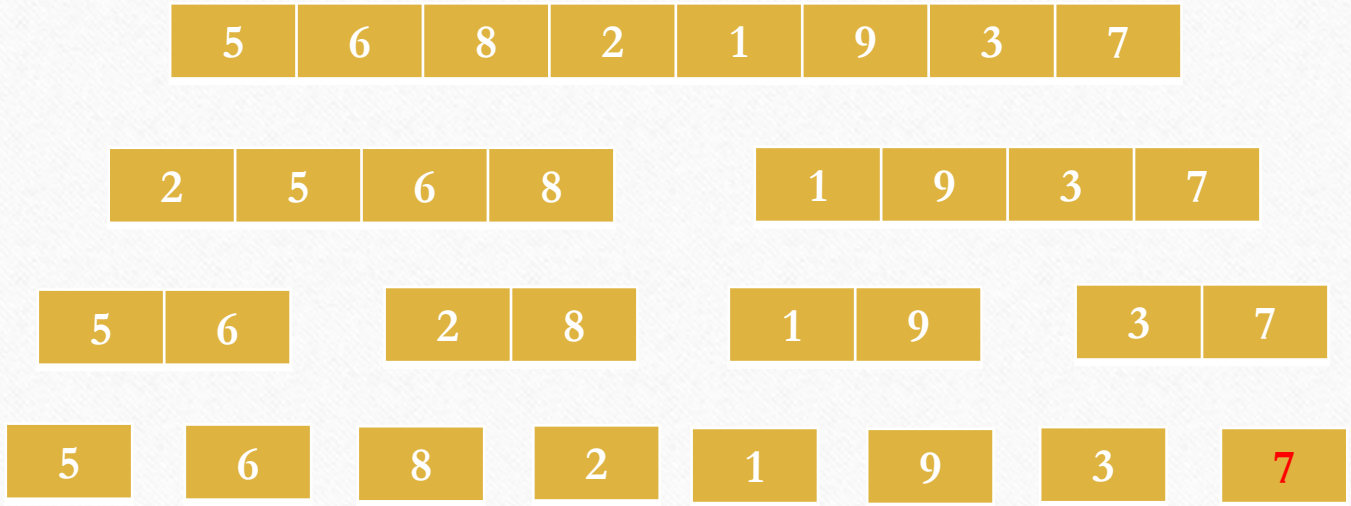

5 6 8 2 1 9 3 7
2 5 6 8
1 9 3 7
5 6
2 8
1 9
5
6
8
2
1

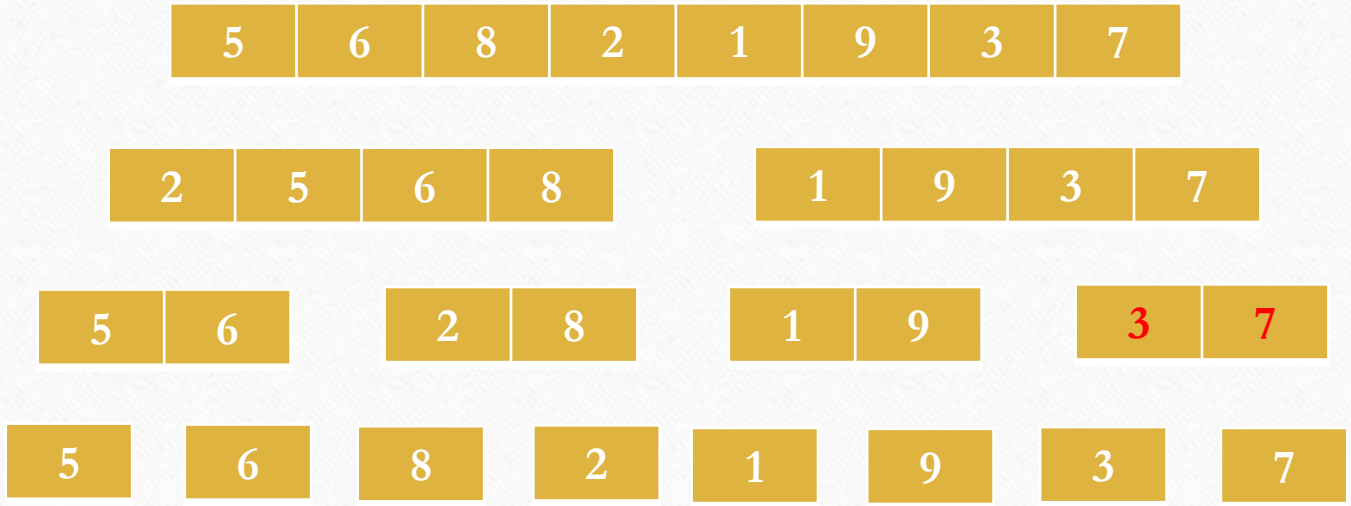

5 6 8 2 1 9 3 7
2 5 6 8
1 9 3 7
5 6
2 8
1 9
5 6 8 2 1 9

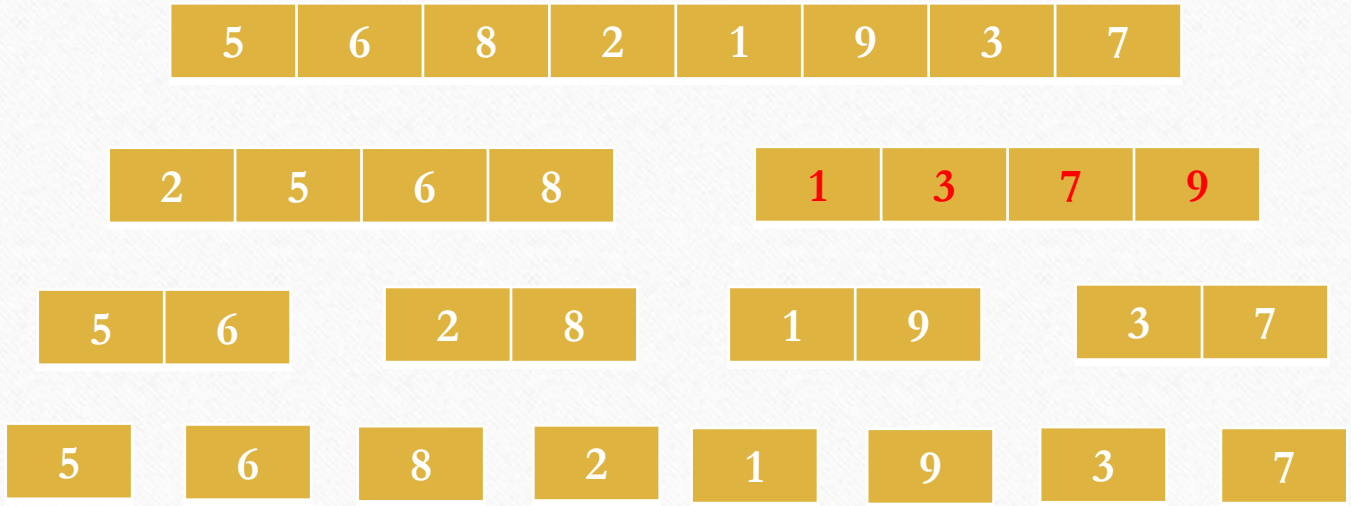

5 6 8 2 1 9 3 7
2 5 6 8
1 9 3 7
5 6
2 8
1 9
5
6
8
2
1
9

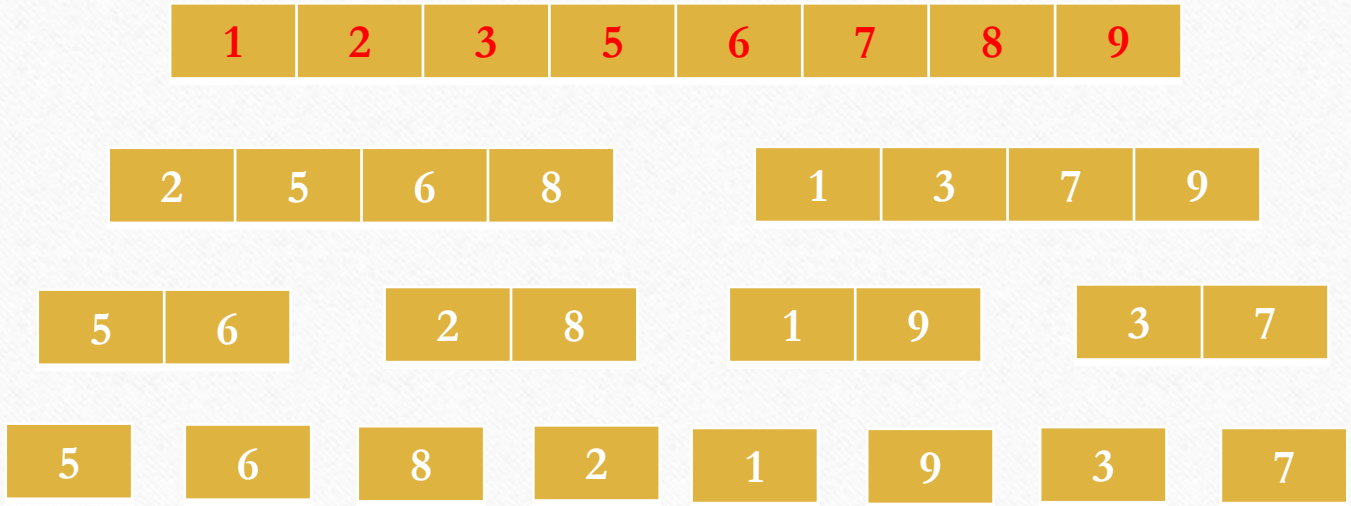

5 6 8 2 1 9 3 7
2 5 6 8
1 9 3 7
5 6
2 8
1 9
3 7
5
6
8
2
1
9


5 6 8 2 1 9 3 7
2 5 6 8
1 9 3 7
5 6
2 8
1 9
3 7
5 6 8 2 1 9 3


5 6 8 2 1 9 3 7
2 5 6 8
1 9 3 7
5 6
2 8
1 9
3 7
5 6 8 2 1 9 3 7


5 6 8 2 1 9 3 7
2 5 6 8
1 9 3 7
5 6
2 8
1 9
3 7
5
6
8
2
1
9
3
7


5 6 8 2 1 9 3 7
2 5 6 8
1 3 7 9
5 6
2 8
1 9
3 7
5
6
8
2
1
9
3
7


1 2 3 5 6 7 8 9
2 5 6 8
1 3 7 9
5 6
2 8
1 9
3 7
5 6 8 2 1 9 3 7

比較

- 自己寫的(以下稱版本**A**)
- Visual Studio反組譯C/C++產生的(以下稱版本**B**)

呼叫函式後的初始化

- 版本**A**(右上圖)只有將ebp儲存在堆疊中

```
MergeSort PROC
    push ebp
    mov ebp,esp
    mov eax,[ebp+12] ; eax=L+1
    inc eax
    mov ebx,[ebp+8] ; ebx=R
```

- 而版本**B**(右下圖) 為了保護資料不被意外執行，多了一些清空記憶體的指令

```
00C022A0  push        ebp
00C022A1  mov         ebp,esp
00C022A3  sub         esp,0FCh
00C022A9  push        ebx
00C022AA  push        esi
00C022AB  push        edi
00C022AC  lea         edi,[ebp-0FCh]
00C022B2  mov         ecx,3Fh
00C022B7  mov         eax,0CCCCCCCCh
00C022BC  rep stos    dword ptr es:[edi]
```

呼叫函式後的初始化

- 這四行是指，是將一部份的區塊設為0xCC，0xCC代表中斷點(int 3)。
- 中斷點常用在debug上面

```
00C022AC  lea           edi,[ebp-0FCh]
00C022B2  mov           ecx,3Fh
00C022B7  mov           eax,0CCCCCCCCh
00C022BC  rep stos      dword ptr es:[edi]
```

```
00C02417  int           3
00C02418  int           3
00C02419  int           3
00C0241A  int           3
00C0241B  int           3
00C0241C  int           3
00C0241D  int           3
00C0241E  int           3
00C0241F  int           3
00C02420  int           3
00C02421  int           3
00C02422  int           3
00C02423  int           3
00C02424  int           3
00C02425  int           3
00C02426  int           3
00C02427  int           3
00C02428  int           3
00C02429  int           3
```

執行位置跳轉

- 版本**A**會傳入自己設的標籤(例如右上圖的MQuit)

```
; if(L+1)<=R return;
cmp eax,ebx
jge MQuit
```

- 版本**B**則是傳遞位置(例如右下圖的main+0D5h)

```
00C0250D  mov         dword ptr [ebp-24h],eax
00C02510  mov         eax,dword ptr [ebp-24h]
00C02513  cmp         eax,dword ptr [n]
00C02516  jge         main+0D5h (0C02545h)
```

讀取傳入參數

- 版本**A**會使用$ebp + N$
- 版本**B**則直接利用$dword\ ptr$ [變數名稱]

```
    ; M=(L+R)>>1
    add ebx,[ebp+12]
    shr ebx,1
    push ebx
```

```
    if (R - L <= 1)return;
00C022BE  mov         eax,dword ptr [R]
00C022C1  sub         eax,dword ptr [L]
00C022C4  cmp         eax,1
00C022C7  jg          sol+2Eh (0C022CEh)
00C022C9  jmp         sol+15Fh (0C023FFh)
```

B版本缺點

- 不一定能產生出最少行的code

```
    if (R - L <= 1)return;
00C022BE  mov         eax,dword ptr [R]
00C022C1  sub         eax,dword ptr [L]
00C022C4  cmp         eax,1
00C022C7  jg          sol+2Eh (0C022CEh)
00C022C9  jmp         sol+15Fh (0C023FFh)
    int M = (R + L) / 2;
00C022CE  mov         eax,dword ptr [R]
00C022D1  add         eax,dword ptr [L]
00C022D4  cdq
00C022D5  sub         eax,edx
00C022D7  sar         eax,1
00C022D9  mov         dword ptr [M],eax
```

總結

- 利用Visual Studio反組譯的code較系統化，更方便
- 自己寫的較彈性