Homework #8 (1)

- Write a function called **NumSort** to sort an integer array from the smallest to the biggest.
- Two arguments will be passed into your function by the **APCS rule**
  - **Array size**
  - **The address of the first element in array**
- The return value of the **NumSort()**: the address of the result array.

Homework #8 (2)

- **Function: NumSort**
  - A result array in which each element is sorted from the smallest to the biggest. (原來的integer array沒有被修改，只是讀取原integer array，排序好的結果存放於result array)
  - 在NumSort function裡，呼叫C standard library所提供的malloc()來配置result array的記憶體空間

Homework #8 (3)

- Homework #7：
  - call_numsort.c => **以C語言撰寫**，設定欲排序的資料，呼叫ARM組合語言寫成的NumSort function。
  - numsort.s => ARM組合語言寫成的NumSort function。
  - **透過APCS規範來傳遞資料。**

Homework #8 (4)

- Homework #8：
  - call_numsort.c => **以C語言撰寫**，設定欲排序的資料，呼叫ARM組合語言寫成的NumSort function。
  - numsort.s => ARM組合語言寫成的NumSort function。
    - NumSort function裡，透過呼叫C所提供的malloc function來配置result array的位址。
  - output.s => ARM組合語言，利用semihosting寫成的FileOutput function。
    - 取代原來在call_numsort.c裡，使用printf()輸出排序好的資料。在這個作業裡，我們改呼叫FileOutput()，將排序好的資料寫到檔案: sort_result.txt。
  - **透過APCS規範來傳遞資料。**

Homework #8 (5)

- **call_numsort.c：以C語言撰寫**，設定欲排序的資料，呼叫ARM組合語言寫成的**NumSort** function。

```
extern int* NumSort(int, int*);

int main(void)
{
    int* result;

    /* initial a integer array */

    /* call NumSort function */
    result = NumSort(array_size, array_address);

    /* Output the result to the file: sort_result.txt*/
    FileOutput(…);

    return 0;
}
```

輸出部分，請使用FileOutput將所排序好的資料輸出至檔案sort_result.txt，每個integer由一個空格隔開

Homework #8 (6)

- 檔案output.s裡包含FileOutput函式
- int FileOutput(char* str)
  - str: 欲輸出字串的位址
  - 傳回值: 0=>輸出成功
  - 使用semihosting實作
  - 將字串輸出至檔案sort_result.txt

File I/O: Write (Semihosting)

- Write a file
  - AngelSWI_Reason_Write = 0x05
  - 3 parameters

File descriptor
Address of the string
Length of the string

  - **Return: r0 (0=> success)**

Homework #8 (7)

- 提醒

```
extern int* NumSort(int, int*);

int main(void)
{
    int* result;

    /* initial a integer array */

    /* call NumSort function */
    result = NumSort(array_size, array_address);

    <把欲輸出的字串準備好，數字與數字間要有一個空格>
    /* Output the result to the file: sort_result.txt*/
    FileOutput(…);

    return 0;
}
```

Use malloc() Function

- R0: the size of memory space (byte)

```
int* ptr = malloc(100);
```

```
…
mov    r0, 100
bl     malloc
…
```

How to Compile Your Program?

- %arm-elf-gcc –g call_numsort.c numsort.s output.s –o \ call_numsort.exe

Homework #8 (8)

- Program should be assembled and linked by gcc (ARM-ELF format)
  - 使用於作業一所編譯完成的cross compiler與cross binutils
- Program should be executed under **GDB ARM simulator**
- 程式中應有適當的說明（註解）
- You should turn in to **ECOURSE**
  - "**README**" file: 文字檔，描述你程式的內容、如何編譯程式、如何執行你的程式
  - Your ARM assembly codes，**檔名為：numsort.s、output.s**
  - A C program which uses your sorting procedure to demo your sorting algorithm，**檔名為：call_numsort.c**
  - Any file needed in your work (ex: Makefile)
  - **Deadline: December 27 (Sunday), 2015**