Homework #6

- Computation: max pooling

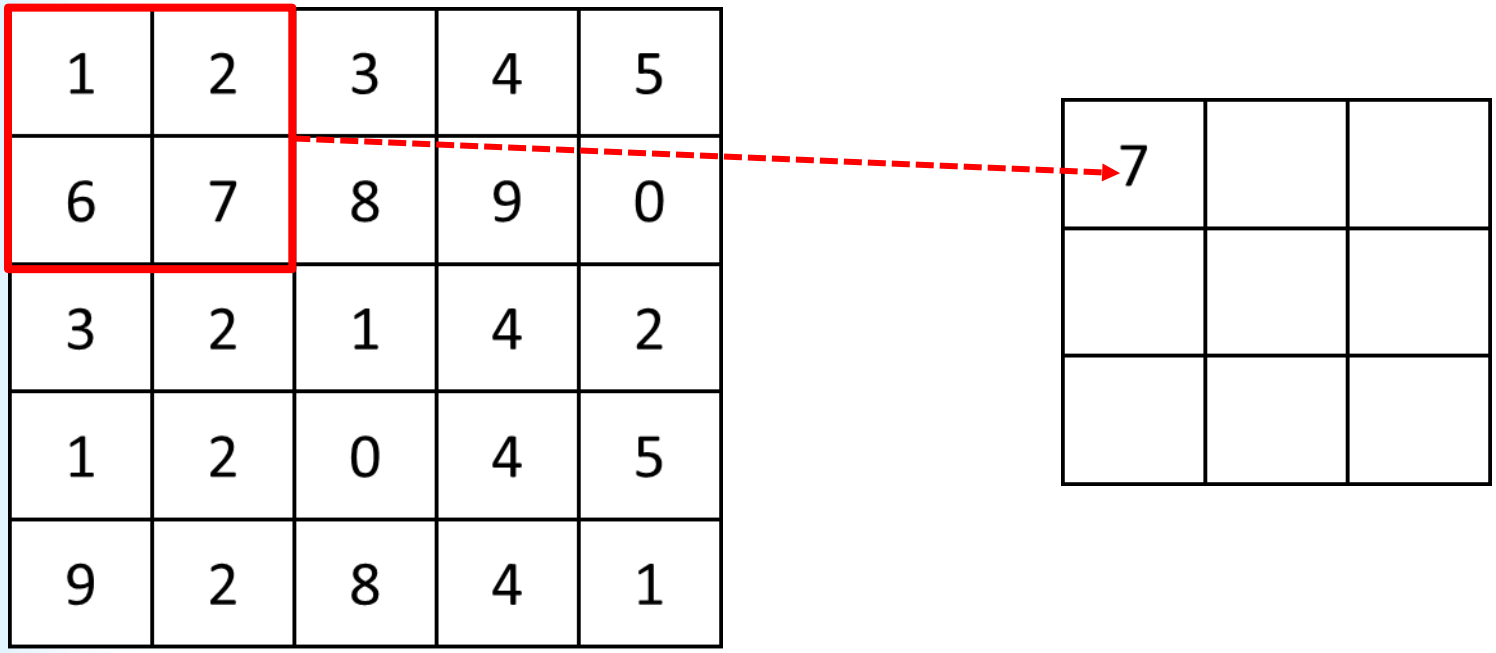
1	2	3	4	5
6	7	8	9	0
3	2	1	4	2
1	2	0	4	5
9	2	8	4	1

Input: 5x5

Kernel: 2x2
Stride: 2

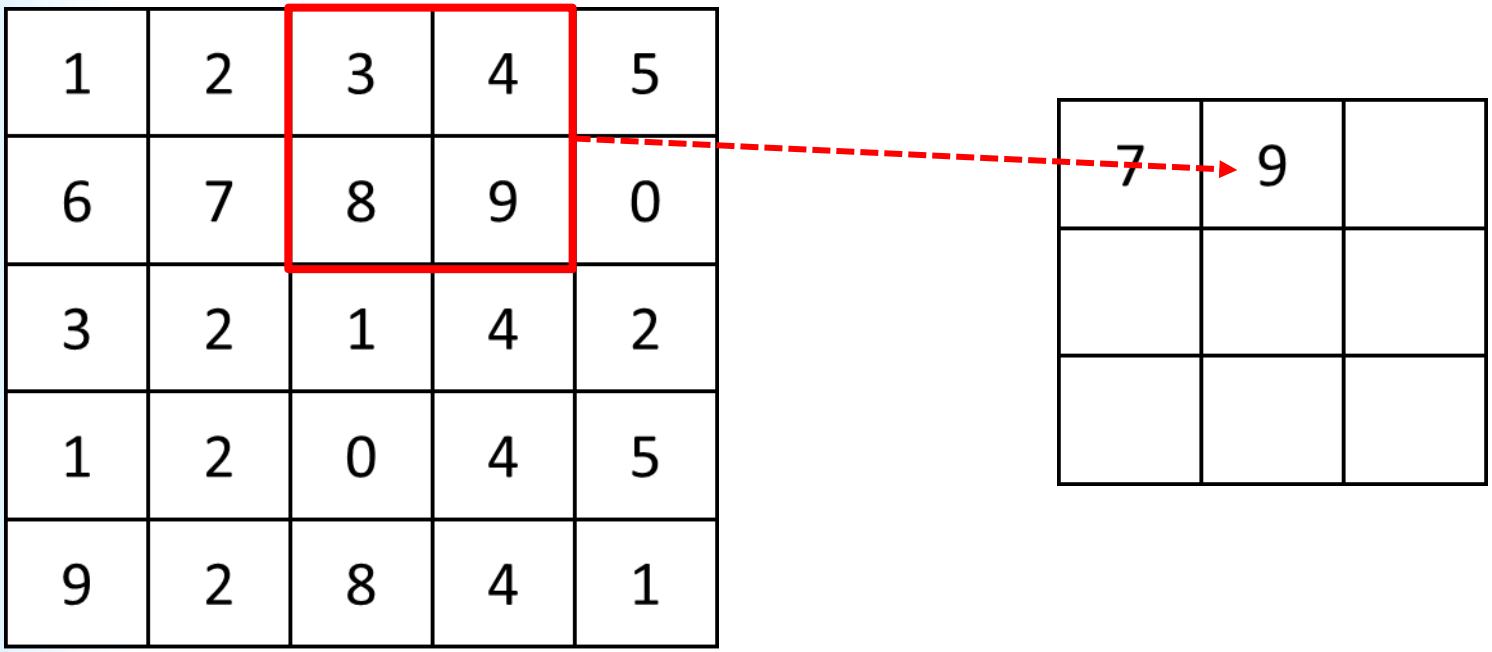
Computation

Kernel: 2x2
Stride: 2



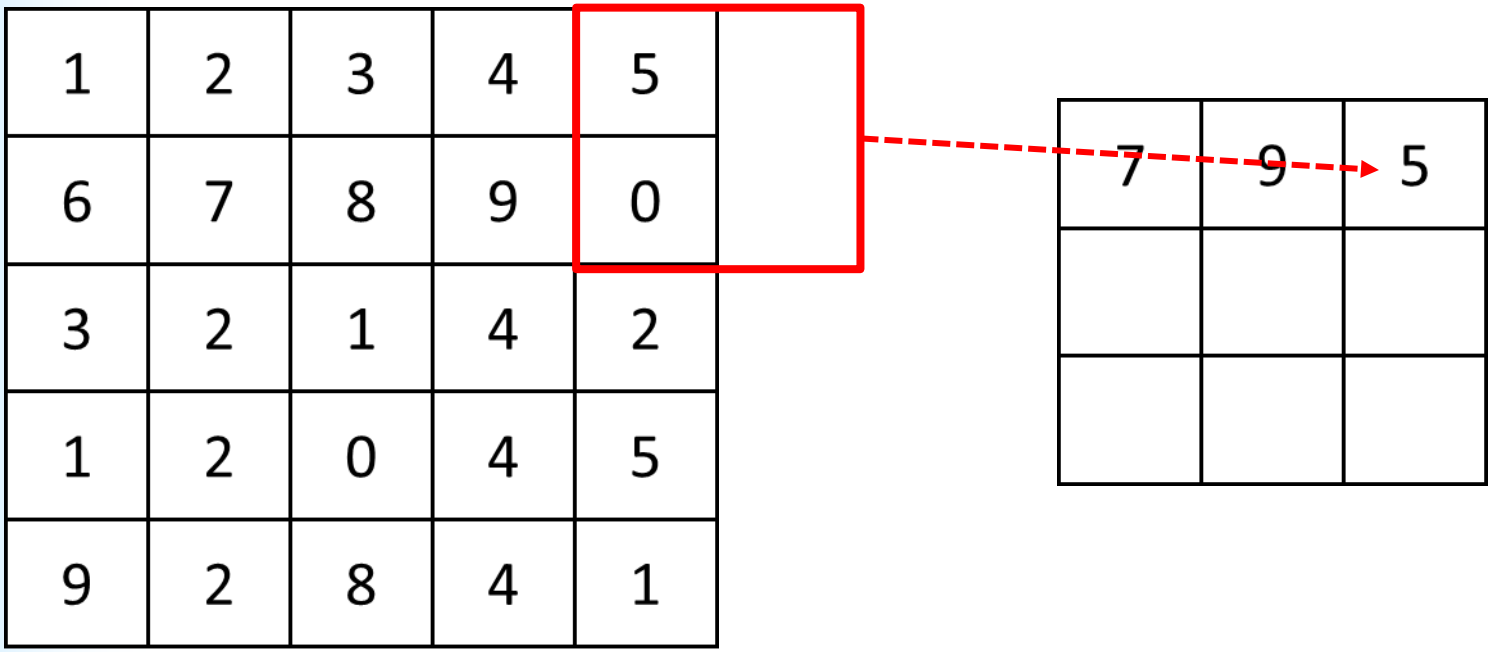
Computation

Kernel: 2x2
Stride: 2



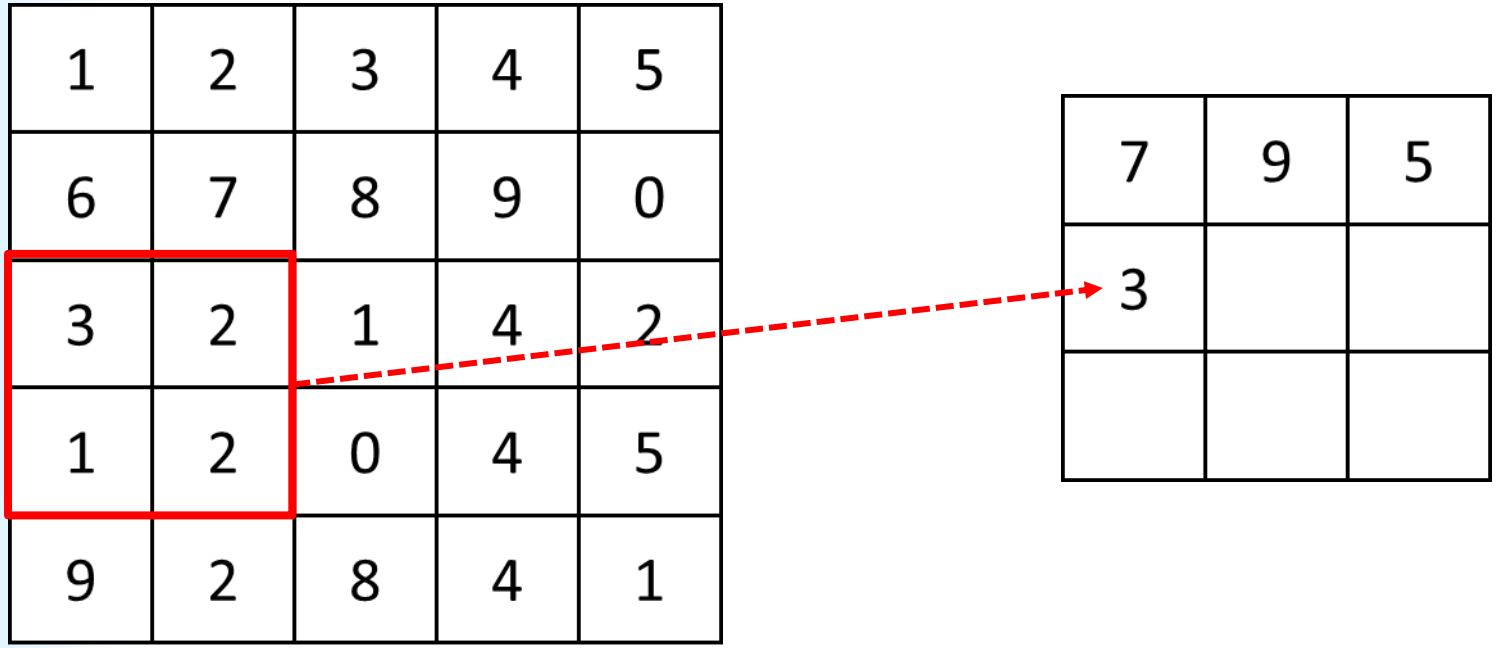
Computation

Kernel: 2x2
Stride: 2



Computation

Kernel: 2x2
Stride: 2



Computation

Kernel: 2x2
Stride: 2

1	2	3	4	5	
6	7	8	9	0	
3	2	1	4	2	
1	2	0	4	5	
9	2	8	4	1	

Max pooling

7	9	5
3	4	5
9	8	1

- Use ARM assembly to write a function called **maxPool** that does the max pooling. **(請參閱作業4)**
- Function **maxPool**: 4 parameters (**遵守APCS規則**)
  - Address of the input matrix (n x m, n>=2, m>=2)
  - Number of rows of the input matrix
  - Number of columns of the input matrix
  - Address of the result matrix
- Function **maxPool**: no return value (**遵守APCS規則**)

Input matrix為 n x m matrix. (n>=2, m>=2)

maxpool.s

hw6_test.c

```
int main(void)
{

    …
    … = maxPool( … );
    …


    return 0;

}
```

參數傳遞

- Address of the input matrix
- Number of rows of the input matrix
- Number of columns of the input matrix
- Address of the result matrix

maxPool function

maxpool.s

```
        .section .text
        .global maxPool
        .type maxPool,%function

maxPool:
        /* function start */
        mov ip, sp
        push {r4-r10, fp, ip, lr, pc}
        sub fp, ip, #4

        /* --- begin your function --- */
        /* 傳入值會放在r0, r1, r2, r3 */

        /* DO max pooling */

        /* --- end of your function --- */

        /* function exit */
        sub     sp, fp, #40
        ldmfd   sp, {r4-r10, fp, sp, lr}
        bx      lr
        .end
```

請留意callee saved registers

Do max pooling

maxpool.s

```
        .section .text
        .global maxPool
        .type maxPool,%function

maxPool:
        /* function start */
        mov ip, sp
        push {r4-r10, fp, ip, lr, pc}
        sub fp, ip, #4

        /* function exit */
        sub     sp, fp, #40
        ldmfd   sp, {r4-r10, fp, sp, lr}
        bx      lr
        .end
```

請留意callee saved registers

中間的程式碼不應該把r11~r15 register 當成 general-purpose register

hw6_test.c

- 準備input matrix
- 透過malloc 索取 result matrix 的記憶體空間
- 呼叫maxPool()
  - Address of the input matrix
  - Number of rows of the input matrix
  - Number of columns of the input matrix
  - Address of the result matrix
- 輸出output the result matrix (透過 printf 輸出)

1	2	3	4	5
6	7	8	9	0
3	2	1	4	2
1	2	0	4	5
9	2	8	4	1

Kernel: 2x2 Stride: 2

Max pooling

7	9	5
3	4	5
9	8	1

輸出output result matrix

(透過 printf 輸出)

```
7 9 5
3 4 5
9 8 1
```

How to Compile Your Program?

```
$ arm-none-eabi-gcc -g -O0 hw6_test.c maxpool.s -o
hw6.exe
```

Homework #6 (1)

- Program should be assembled and linked by gcc
  - 使用於作業一所安裝完成的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）
- 程式應遵守APCS規則
- You should turn in to **ECOURSE2**
  - "**README.txt**" 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 (特別註明你的執行環境是否為Mac系統)
  - Your ARM assembly procedure，**檔名為：maxpool.s**
  - A C program which uses your function，**檔名為：hw6_test.c**
  - Makefile
  - Any file needed in your work

Homework #6 (2)

- 請勿繳交【利用編譯器所自動產生的組合語言程式】
- 請勿抄襲
- 請將欲繳交的檔案壓縮成 <hw6_學號.tar.xz>，上傳壓縮檔
- **Deadline: November 24 (Sunday), 2024**