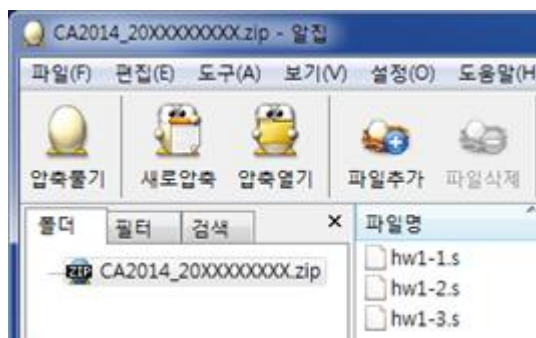# Lab #1

ICE3003-43: Computer Architecture

Parallel Architecture and Programming Lab

Chungha Sung (sch8906@gmail.com)

Yunkeuk Kim (giocafe87@gmail.com)

Overview

- Download the attached template files,
  and translate C code into MIPS assembly code.
- Save your code without changing the file names,
  and compress them into a single zip.
- File name template: **CA2014_{your student ID}.zip**

- Upload the compressed file onto **iCampus.**
  - Due date   : **4/8 (Tue) 4:30 PM** before class.
  - Penalty   : 20% points off per one late day
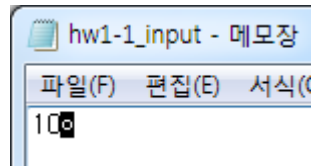   based on the time stamp on iCampus.

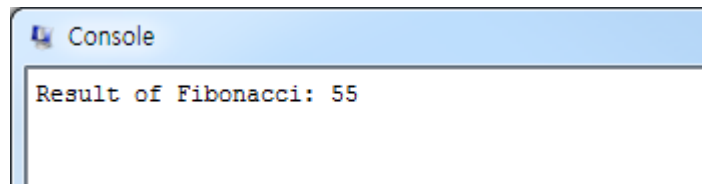# Lab #1-1 (Fibonacci)

Principle

$a_n = a_{n-1} + a_{n-2}$

$(a_0 = 0, a_1 = 1, n \geq 2)$

Input

10

Output

Example

0	1	1	2	3	5	8	13	21	34	**55**
a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	**a_{10}**

Lab #1-1 (Fibonacci)

C code

```
int fib (int input) {
    if (input == 0)
        return 0;
    else if (input == 1)
        return 1;
    else
        return fib (input-1) + fib (input-2);
}
```
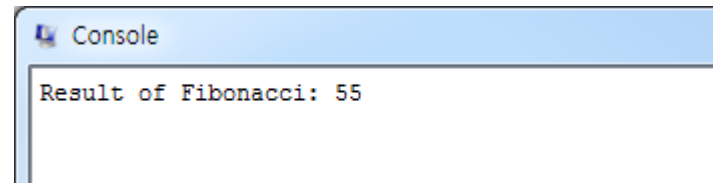
Fibonacci

Print out the Nth fibonacci number.
You should complete the code for
"hw1-1.s".

Input

hw1-1_input.txt

Output

Console
Result of Fibonacci: 55

Lab #1-1 (Fibonacci)

hw1-1.c (reference code)

```
int fib (int input) {
    if (input == 0)
        return 0;
    else if (input == 1)
        return 1;
    else
        return fib (input-1) + fib (input-2);
}
```

hw1-1.s (please fill in)

```
        .data
Input:      .asciiz         "hw1-1_input.txt"
output:     .asciiz         "Result of Fibonacci: "
buffer:

                                    :

fib:


##################################################
# You are allowed to modify this part of the code only: START
##################################################


        ## FILL IN YOUR CODE HERE!! ##


##################################################
# You are allowed to modify this part of the code only: END
##################################################


    jr      $ra
```

Lab #1-2 (Maxheap)

Principle

of Parent = # of Child / 2

Input

hw1-2_input - 메모장

파일(F) 편집(E) 서식(G)
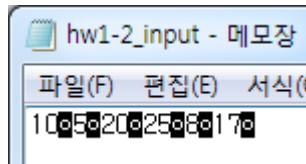
10 5 20 25 8 17

10 → 5 → 20 → 25 → 8 → 17

Output

Console

```
25
---------------
20  17
---------------
5  8  10
```
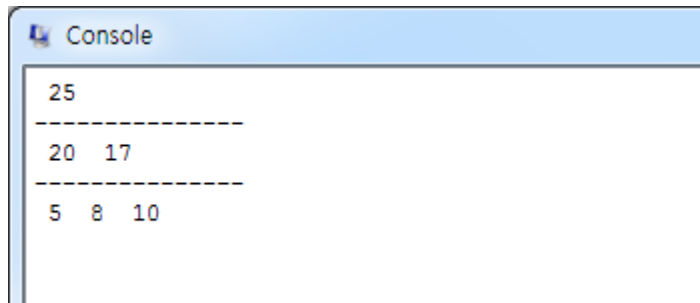
Example
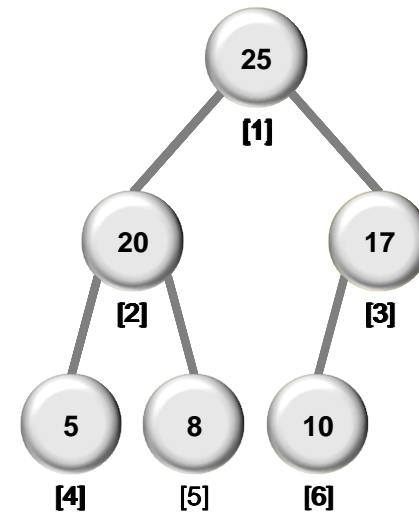
	[0]	[1]	[2]	[3]	[4]	[5]	[6]	...
heap	-	25	20	17	5	8	10	

Lab #1-2 (Maxheap)

C code

```
int insert_heap (int curr, int input) {
    int i;
    i = curr;
    heap[curr] = input;
    while ((i != 1) && (heap[i] > heap[i/2])) {
        swap (&heap[i], &heap[i/2]);
        i = i/2;
    }
}

void swap (int *a, int *b) {
    int t = *a;
    *a = *b;
    *b = t;
}
```
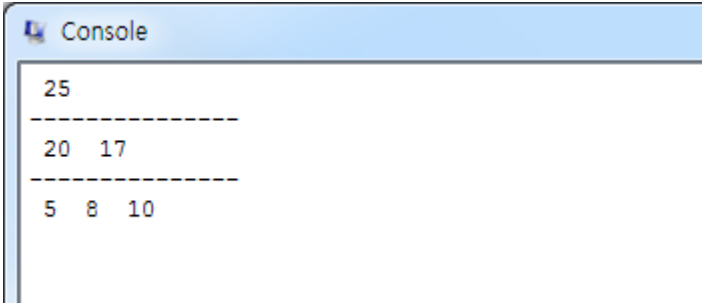
Maxheap

Insert the numbers in the input file into heap in an order of Maxheap algorithm. You should complete the code for "hw1-2.s".

Input

hw1-2_input.txt

Output

Lab #1-3 (Anagram)

Principle

$N = {}_nP_1 + {}_nP_2 + \ldots + {}_nP_n$

Input

hw1-3_input - 메모장
파일(F) 편집(E) 서식(O
abc

abc

Output

Console

```
a  1
b  2
c  3
ab  4
ac  5
ba  6
bc  7
ca  8
cb  9
abc  10
acb  11
bac  12
bca  13
cab  14
cba  15
```

Example

$_3P_1$ $_3P_2$ $_3P_3$

a — b — ab — c — abc
a — c — ac — b — acb
b — a — ba — c — bac
b — c — bc — a — bca
c — a — ca — b — cab
c — b — cb — a — cba
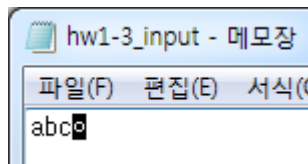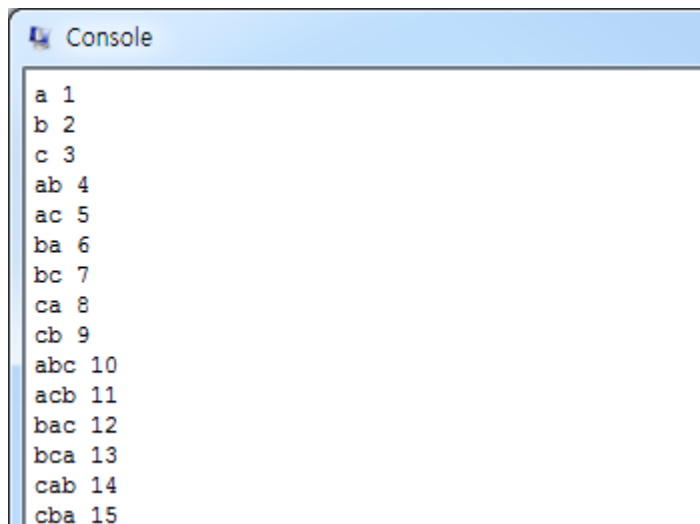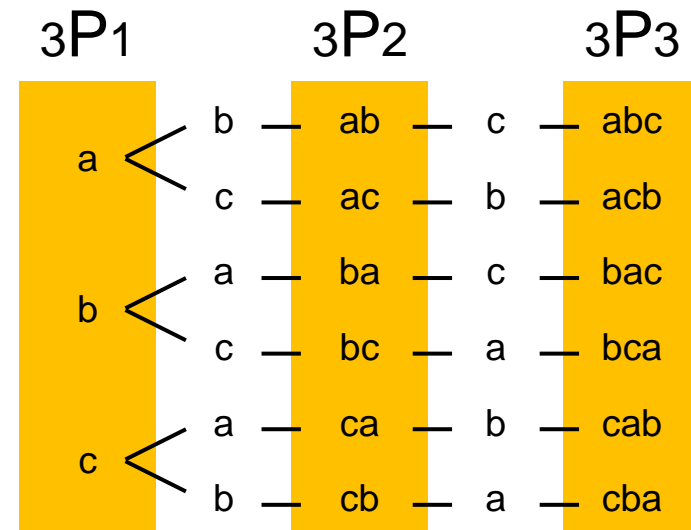
$\therefore N = {}_3P_1 + {}_3P_2 + {}_3P_3$

Lab #1-3 (Anagram)

C code

```
void execute_anagram (int size) {
    int i;
    for (i = 1; i <= size; i++)
        permutation (size, i);
}

void permutation (int n, int r) {
    int i, j;
    if (r==0)
        print();
    else {
        for (i = 0; i < n; i++) {
            for (j = 0; j < cnt; j++) {
                if (i == array_index[j])
                    break;
            }
            if (j == cnt) {
                array_index[cnt] = i;
                print_array[cnt] = original_array[i];
                cnt++;
                permutation (n, r-1);
                cnt--;
            }
        }
    }
}
```
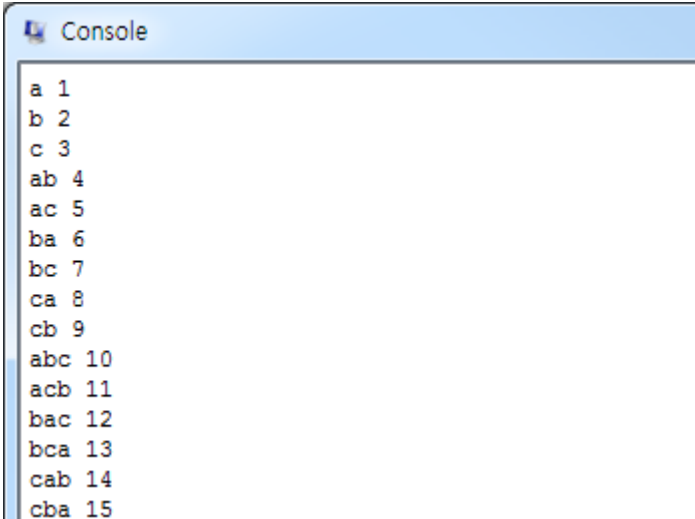
Anagram

Read the characters in the input file
and make an anagram. And then,
print out result. You should complete
the code for "hw1-3.s".

Input

hw1-3_input.txt

Output

```
Console
a 1
b 2
c 3
ab 4
ac 5
ba 6
bc 7
ca 8
cb 9
abc 10
acb 11
bac 12
bca 13
cab 14
cba 15
```

Execution Tips (Windows)

- **Run Visual Studio**
- **Make a new project and add the C code.**
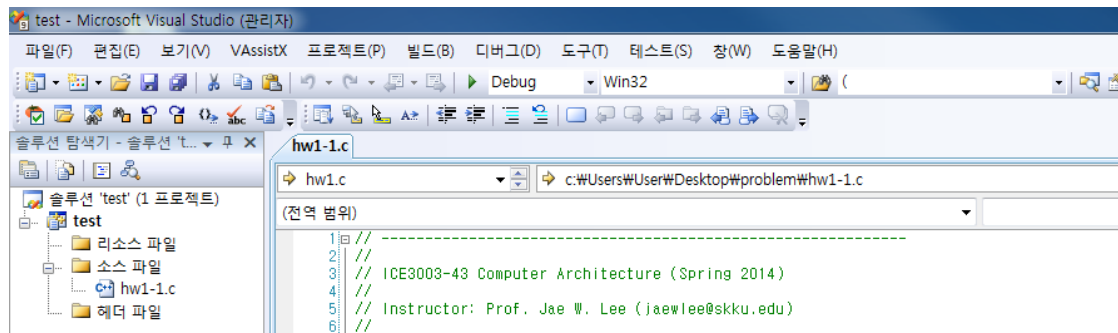
- **Compile (Ctrl + F7)**
- **Build (F7)**
- **Run (Ctrl + F5)**

(Ctrl + Alt + B)

Execution Tips (Windows)

** You have to copy input files into project path.
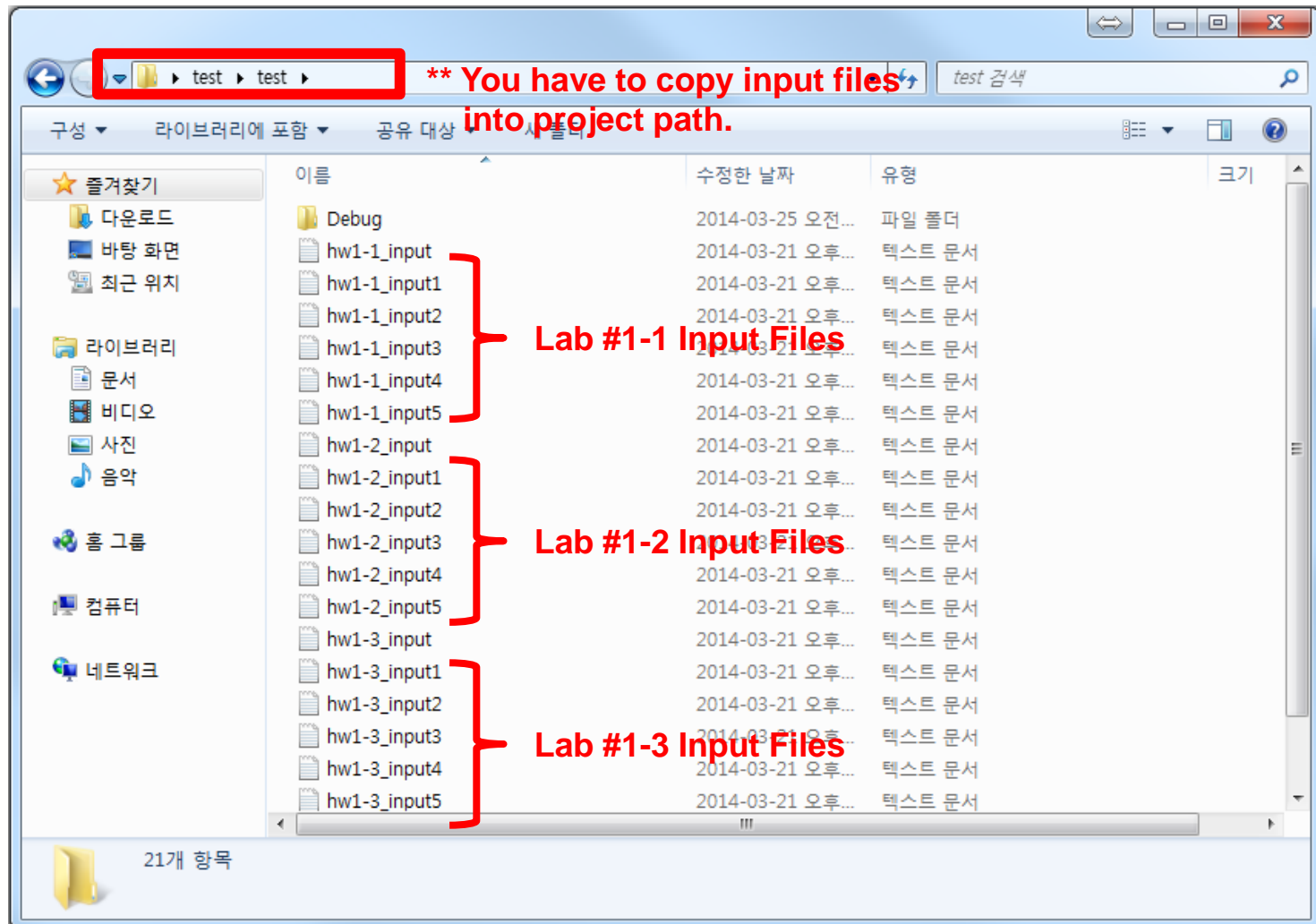
** We will test your codes with the released input files only.

Execution Tips (Windows)

- **Download & Install & Run QtSpim**

  http://sourceforge.net/projects/spimsimulator/files/

- **Load assembly file**

- **Run assembly file**

Success!!

Execution Tips (Windows)

Execution Tips (Windows)

- **If you want to test with the other input files,**

  you have to modify the code.

  hw1-1.c

  hw1-1.s

- **Modify the input file name in code.**

  hw1-1.c

  ```
  // file open
  fp = fope("hw1-1_input.txt", "r");
  ```

  hw1-1.s

  ```
  .data
  input   :  .asciiz    "hw1-1_input.txt"
  output  :  .asciiz    "Result of Fibonacci: "
  ```

- **And then, you should re-compile and re-run the files.**

Execution Tips (Ubuntu)

- **Go to the Lab #1 download path**

  ex) $ cd ~/Desktop/problem
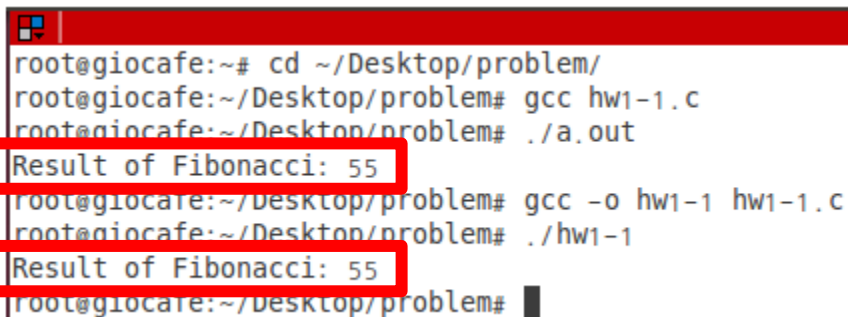
- **Compile & Run**

  ex1) $ gcc hw1-1.c

  $ ./a.out

  ex2) $ gcc –o "name" hw1-1.c

  $ ./ "name"

Execution Tips (Ubuntu)

- **Install spim.**

  $ sudo apt-get install spim

- **Go to the Lab#1 path and run spim like this.**

  $ spim -f hw1-1.s

Execution Tips (Ubuntu)

Execution Tips (Ubuntu)

- **If you want to test with the other input files,**

  you have to modify the code.

  ```
  $ vi hw1-1.c
  $ vi hw1-1.s
  ```

- **Press 'i' or 'a' and modify the input file name in fopen fuction.**

  hw1-1.c
  ```
  28
  29      // file open
  30      fp = fope ("hw1-1_input.txt"  "r");
  31
  ```

  hw1-1.s
  ```
  27 .data
  28 input    : .asciiz     "hw1-1_input.txt"
  29 output   : .asciiz     "Result of Fibonacci: "
  ```

- **Press ESC, and ":wq".**

  And then, you should re-compile and re-run the files.

Grading Guidelines

• **Working Correctly**

   - Your MIPS code outputs the same result as the reference C code.

   - We don't care about performance or algorithms you use.

   ** **You are allowed to modify only those part labeled**

     with **"## FILL IN YOUR CODE HERE!! ##"** in your MIPS code.

   ** **You are allowed to modify the name of the input files**

     to try different inputs for verifying or debugging.

       ex) hw1-1_input1, …, hw1-1_input5, …, hw1-3_input1, …, hw1-3_input5

   ** **But, you NEVER modify the contents of the input files**

     even though some characters in file are broken. Never mind.

• **Comments**

   - Insert appropriate comments to help TAs understand your code.

   - Please exercise your common sense. TAs are reasonable guys.

Grading Guidelines

Code	Score (pts)	Description (pts)
Fibonacci	30	Working correctly: 25 Comments: 5
Maxheap	30	Working correctly: 25 Comments: 5
Anagram	40	Working correctly: 35 Comments: 5

Tutorials & Questions

- **QtSpim & SPIM tutorial will be given by TAs this Friday.**
    - When   : **3/28 (Fri) 3 PM**
    - Where : **400118 (This room)**
    - It will be videotaped, and your attendance is totally optional.

- **If you have a question, consult the TAs**
  during the office hours or by emails.
    - Office hours   : 3~4 PM (Tue, Fri)
    - Chungha Sung : sch8906@gmail.com
    - Yunkeuk Kim   : giocafe87@gmail.com

Thank you