EEE3050 Theory on Computer Architectures (Spring 2017)

# HW1: MIPS Assembly

2017.3.27 (MON)

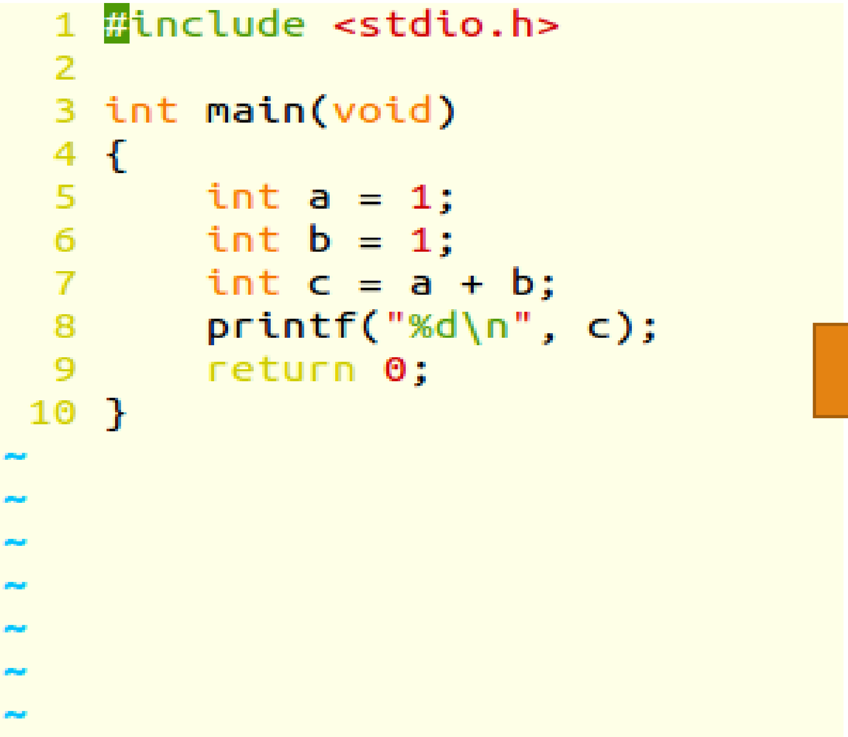
TA 이규선 / 안민우

# What to do?

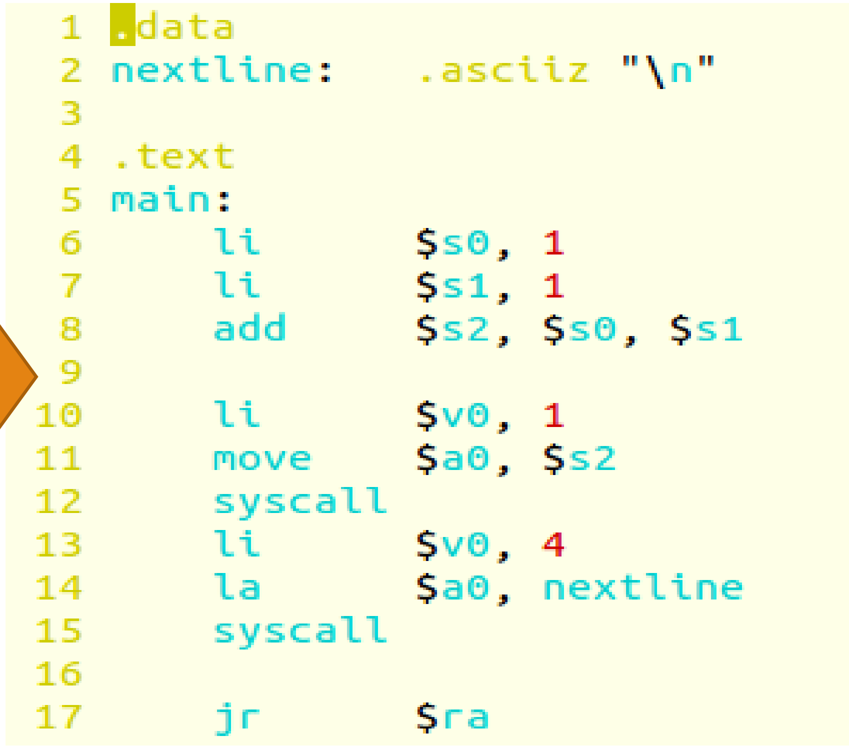
**BECOME**
**HUMAN COMPILER!!**

# What to do?

- Converting given C code to “MIPS” assembly.



```c
#include <stdio.h>

int main(void)
{
	int a = 1;
	int b = 1;
	int c = a + b;
	printf("%d\n", c);
	return 0;
}
```



```
.data
nextline:	.asciiz "\n"

.text
main:
	li	$s0, 1
	li	$s1, 1
	add	$s2, $s0, $s1

	li	$v0, 1
	move	$a0, $s2
	syscall
	li	$v0, 4
	la	$a0, nextline
	syscall

	jr	$ra
```

# Enviornment

- Windows OS
- QtSpim (GUI) simulator
  - You can download a various version of QtSpim (Mac, Win, Linux) at http://sourceforge.net/projects/spimsimulator/files/
  - SPIM (no GUI) only for Linux (Refer Appendix)
  - How to use QtSpim?
    - There will be a SPIM/QtSpim tutorial on Thursday(3/30) 6PM. It may takes 1 hour.

# Given Files

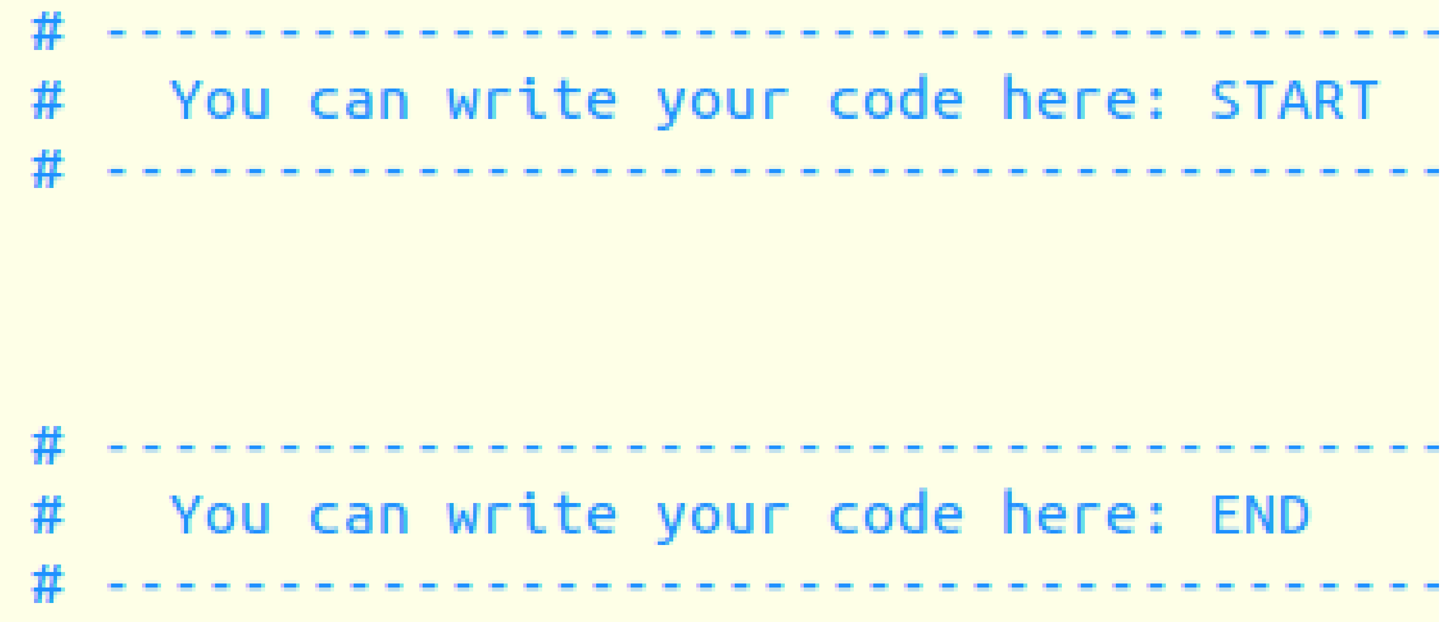
- hw1-#.c (not have to submit)
  - Source C code file
  - You can compile and execute it.
- hw1-#-main.s (not have to submit)
  - MIPS assembly code of a main() function and global variables
  - Do not modify it during homework
- hw1-#-function.s (have to submit)
  - This is what you have to fill.
  - MIPS assembly code of user function.
- hw1-#.input (not have to submit)
  - Input data files

# Editor

- You can edit given files(e.g. *.c, *.s, *.input, *.sh, *.bat) by any text editors.

- But, we recommend to use WordPad(워드패드 in Windows)

# hw1-#-function.s file

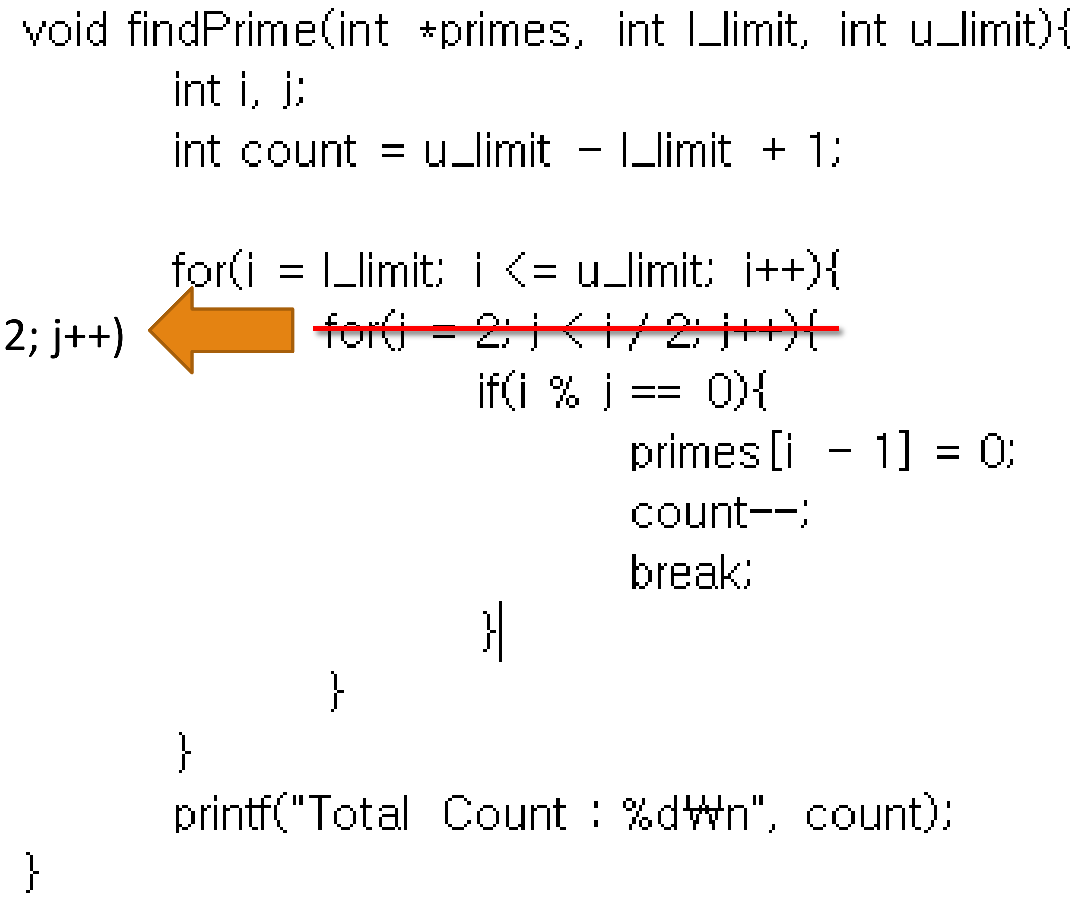


```
# ---------------------------------------
#   You can write your code here: START
# ---------------------------------------




# ---------------------------------------
#   You can write your code here: END
# ---------------------------------------
```

# HW1-1: Find all primes btw two integers

- main.s
    - We will give you MIPS assembly wrapper: hw1-1-main.s
- function.s
    - You must fill in

for(j = 2; j <= i / 2; j++)



```
void findPrime(int *primes, int l_limit, int u_limit){
        int i, j;
        int count = u_limit - l_limit + 1;

        for(i = l_limit; i <= u_limit; i++){
                ~~for(j = 2; j < i / 2; j++){~~
                        if(i % j == 0){
                                primes[i - 1] = 0;
                                count--;
                                break;
                        }
                }
        }
        printf("Total Count : %d₩n", count);
}
```

# HW1-1: Find all primes btw two integers

- Input file format
  - ~~# of testcase is given in first line.~~
  - Separators are ‘ ‘ and ‘\n’
  - First number should be smaller than second one
  - Input numbers are positive integers between 1 and 200

(Input File)

~~1~~

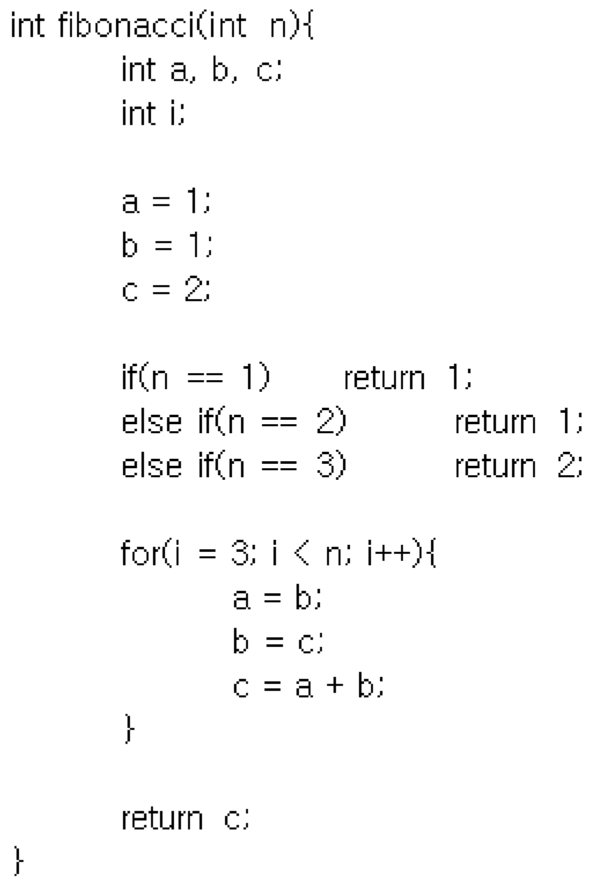
2 100

- Output

```
(Console)
        Prime numbers between 2 and 100 are
        Total Count : 25
        2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
```

# HW1-2: Find n$^{th}$ Fibonacci Number

- main.s
  - We will give you MIPS assembly wrapper: hw1-2-main.s
- function.s
  - You must fill in.
  - This function has return value.



```
int fibonacci(int  n){
        int a, b, c;
        int i;

        a = 1;
        b = 1;
        c = 2;

        if(n == 1)      return 1;
        else if(n == 2)         return 1;
        else if(n == 3)         return 2;

        for(i = 3; i < n; i++){
                a = b;
                b = c;
                c = a + b;
        }

        return c;
}
```

# HW1-2: Find n$^{th}$ Fibonacci Number

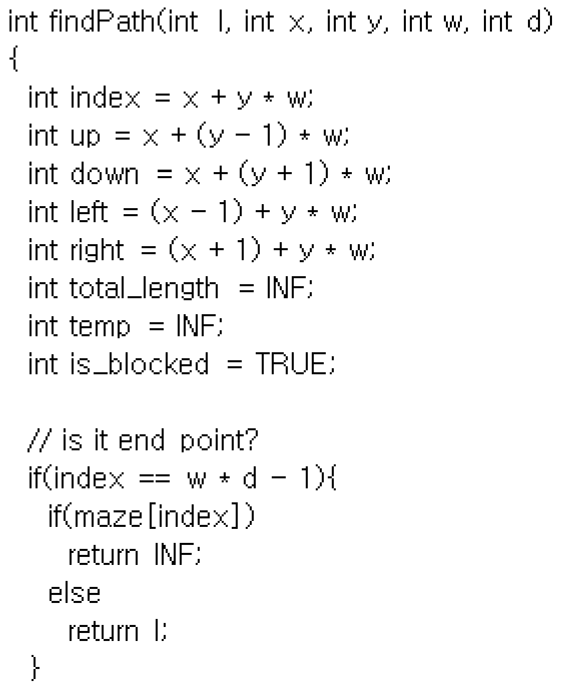
- Input file format
  - ~~# of testcases is given in first line~~
  - From next line, number means “n”
- Output

(Input File)
~~2~~
3
5

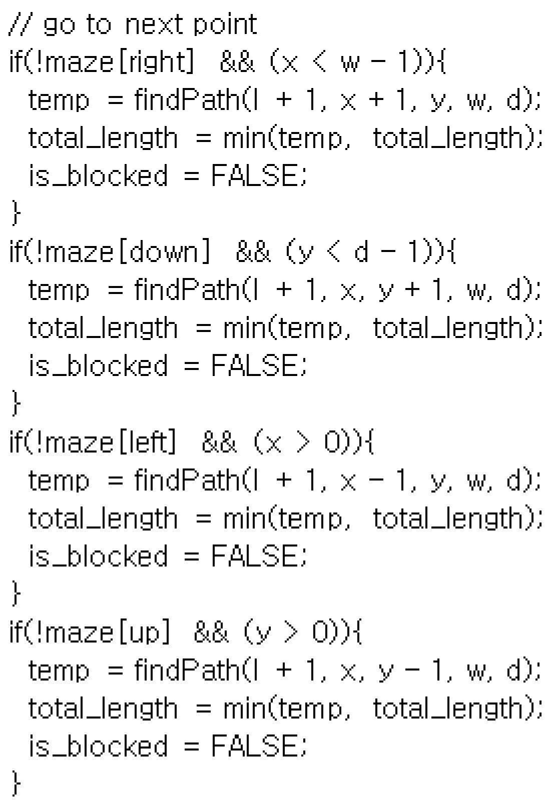
```
(Console)
        3rd fibonacci number is 2
        5th fibonacci number is 5
```

# HW1-3: Maze Solving

- main.s
  - We will give you MIPS assembly wrapper: hw1-3-main.s
- function.s
  - You must fill in
  - This function has 5 arguments
  - Recursive



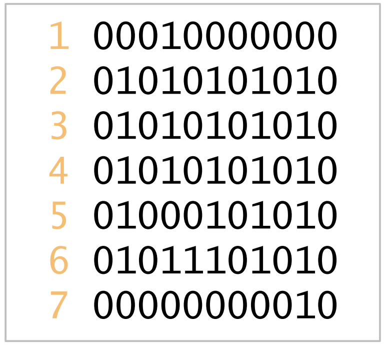
```
int findPath(int  l, int x, int y, int w, int d)
{
  int index = x + y * w;
  int up = x + (y - 1) * w;
  int down  = x + (y + 1) * w;
  int left = (x - 1) + y * w;
  int right  = (x + 1) + y * w;
  int total_length  = INF;
  int temp  = INF;
  int is_blocked  = TRUE;

  // is it end  point?
  if(index ==  w * d - 1){
    if(maze[index])
      return  INF;
    else
      return  l;
  }
```



```
// go to next point
if(!maze[right]   && (x < w - 1)){
  temp  = findPath(l  + 1, x + 1, y, w, d);
  total_length  = min(temp,  total_length);
  is_blocked  = FALSE;
}
if(!maze[down]   && (y < d - 1)){
  temp  = findPath(l  + 1, x, y + 1, w, d);
  total_length  = min(temp,  total_length);
  is_blocked  = FALSE;
}
if(!maze[left]   && (x > 0)){
  temp  = findPath(l  + 1, x - 1, y, w, d);
  total_length  = min(temp,  total_length);
  is_blocked  = FALSE;
}
if(!maze[up]   && (y > 0)){
  temp  = findPath(l  + 1, x, y - 1, w, d);
  total_length  = min(temp,  total_length);
  is_blocked  = FALSE;
}
```

# HW1-3: Maze Solving

- Input file format
  - Total number of 1s, 0s, and ‘\n’ should be unber 400
  - 0 is path and 1 is wall



```
00010000000
01010101010
01010101010
01010101010
01000101010
01011101010
00000000010
```

- Output

(Console)

```
Shortest path length is 25
The maze looks like
w = 11, d = 7
00010000000
01010101010
01010101010
01010101010
01000101010
01011101010
00000000010
```

# Submission

- Compress your three hw1-#-function.s files only(Don't change file name)
  - Without subdirectories
  - YourStudentID.zip
  - YOU MUST FOLLOW THIS FORMAT. If not, your grade …
- Upload your zip file to I-Campus Assignments bulletin
- PLEASE DO NOT COPY. If not, your grade ………….ㅠㅠ
- Due date:

# Tutorial and Skeleton Analysis

- SPIM/QtSpim tutorial
  - When: 3/30(Thur), 6PM
  - Where: TBA(Somewhere in semiconductor building)
  - Attendance is not mandatory.

# Questions

- You are free to ask questions to TA. (Email | Semiconductor Building #400509)