

# Activity 4 Simple shell

## Group: กุ้ง

## Members

1. 6231316921 ณฐมน พลวิชัย
2. 6232035721 แสนยากร เสี่ยงเสนาะ
3. 6232037021 อานนท์ จินดาวงศ์

## Problem 1

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/wait.h>

int main(int argc, char *argv[])
{
    pid_t pid;

    pid = fork();
    if (pid < 0)
    {
        printf("Error : cannot fork\n");
        exit(1);
    }
    else if (pid == 0)
    {
        execvp(argv[1], argv + 1);
    }
    else
    {
        wait(NULL);
        return (0);
    }
}
```

## อธิบาย

เราสามารถรับ arguments ต่าง ๆ ของการเรียกโปรแกรมได้  
จากนั้นเราก็จะใช้ arguments นี้ผ่านเข้าไปใน Process ลูก  
หลังจากที่ execute เสร็จแล้วก็ให้ Process แม่ปิดโปรแกรมลง

## แก้ไข

`argc` คือจำนวน argument ที่ใส่เข้ามาและ `argv` คือ argument ที่ถูกตัดมาแล้วเช่น

```
./problem1 cal -m 11
```

จะได้ `argc = 4` และ `argv` เป็น list ของ string `['./problem1', 'cal', '-m', '11']`

จากนั้นใช้คำสั่ง `execvp()` เพื่อรันคำสั่งใน `argv` ตั้งแต่ตัวที่ 1 เป็นต้นไป ในที่นี้คือ `['cal', '-m', '11']`

## ผลลัพธ์

```
~/Desktop/2110313-OS-SYS-P
> ./problem1 cal -m 11
November 2022
Su Mo Tu We Th Fr Sa
      1  2  3  4  5
 6  7  8  9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30
```

## Problem 2

### แนวคิด

รับคำสั่งจาก terminal ด้วย `fgets()` จากนั้นเอาไป `tokenize()` แล้วใช้คำสั่ง `fork()` แล้วตามด้วย `execvp()`

### แก้ไข

รับสตริงเข้ามาด้วย `fgets()` แล้วแยกออกมาให้เป็น token จากนั้นทำการเช็คค่า token แรกเป็น `exit` หรือไม่ ถ้าใช่ก็หยุดการทำงาน แต่ถ้าไม่ใช่ ก็ fork child process ออกมา และทำการสั่ง `execvp(tokens[0], tokens)` เพื่อรันคำสั่งใน child process

## ผลลัพธ์

```
> gcc -o problem2 ./problem2.c && ./problem2
mysh > cal -m 11
      November 2022
Su Mo Tu We Th Fr Sa
                1  2  3  4  5
 6  7  8  9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30

mysh > ls -l
total 224
-rwxr-xr-x  1 saenyakorn  staff  49592 Feb  4 20:39 problem1
-rw-r--r--  1 saenyakorn  staff    328 Feb  4 20:40 problem1.c
-rwxr-xr-x  1 saenyakorn  staff  50184 Feb  4 21:08 problem2
-rw-r--r--  1 saenyakorn  staff   2293 Feb  4 20:59 problem2.c
mysh > 
```

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>

int tokenize(char *string, char *delimiters, char ***arrayOfTokens);

int main()
{
    int run = 1;

    while (run)
    {
        printf("mysh > ");

        char delim[] = " \t\n";
        char **tokens;
        char string[256];
        int numtokens;
        int i;

        fgets(string, 256, stdin);
        numtokens = tokenize(string, delim, &tokens);

        if (strcmp(tokens[0], "exit") == 0)
        {
            printf("Bye Bye\n");
            return 0;
        }

        pid_t pid = fork();
```

```
if (pid < 0)
{
    printf("Error : cannot fork\n");
    exit(1);
}
// Child process
else if (pid == 0)
{
    execvp(tokens[0], tokens);
}
// Parent process
else
{
    wait(NULL);
}
}
return 0;
}

...
```