ข้อ 2

```c
#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 input[256];
        fgets(input, 256, stdin);

        // Remove newline character from input
        input[strcspn(input, "\n")] = 0;

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

        // 1. use tokenize() function to get command
        numtokens = tokenize(input, delim, &tokens);

        if (strcmp(input, "exit") == 0)
        {
            run = 0;
        }
        else
        {
            if (numtokens > 0)
            {
                // 2. fork a child process
                pid_t pid = fork();

                if (pid < 0)
                {
                    perror("Fork failed: pid < 0\n");
                    exit(1);
                }
                else if (pid == 0)
                {
                    // 3. child use execvp to run command
                    execvp(tokens[0], tokens);
                }
                else
                {
                    // 4. parent call wait() until user enter exit
                    wait(NULL);
                }
            }
        }
    }

    return 0;
}
```

```c
int tokenize(char *string, char *delimiters, char ***arrayOfTokens)
{
    char *token;
    int numtokens;
    int i;
    /* skip the beginning delimiters */
    string += strspn(string, delimiters);
    if ((token = malloc(strlen(string) + 1)) == NULL)
        return -1;
    /* count tokens */
    strcpy(token, string);
    numtokens = 0;
    if (strtok(token, delimiters) != NULL)
        for (numtokens = 1; strtok(NULL, delimiters) != NULL;
             numtokens++)
            ;
    /* create array of pointers to tokens */
    if ((*arrayOfTokens = malloc((numtokens + 1) * sizeof(char *))) ==
        NULL)
    {
        free(token);
        return -1;
    }
    /* fill pointers to tokens into the array */
    if (numtokens == 0)
        free(token);
    else
    {
        strcpy(token, string);
        (*arrayOfTokens)[0] = strtok(token, delimiters);
        for (i = 1; i < numtokens; i++)
            (*arrayOfTokens)[i] = strtok(NULL, delimiters);
        (*arrayOfTokens)[numtokens] = NULL;
    }
    return numtokens;
}
```

```
jackkahod@Jackkahod:~/os-4$ gcc -o ran main.c
jackkahod@Jackkahod:~/os-4$ ./ran
mysh >date
Sat Feb  3 16:56:08 +07 2024
mysh >cal 12 2012
   December 2012
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 31
mysh >exit
jackkahod@Jackkahod:~/os-4$
```

แก้ไขในส่วนของ main โดยการใช้ fgets เพื่อรับข้อมูลจาก user เข้ามาซึ่งมีความยาวไม่เกิน 256 ตัวอักษร และใช้ “input[strspn(input, "\n")] = 0” เพื่อกำจัด \n ออก และจากนั้นใช้ฟังก์ชัน tokenize และนำค่าที่ได้จากฟังก์ชันนี้มาเปรียบเทียบ โดยหากเป็น exit จะทำให้ run = 0 เพื่อหยุดการทำงานของลูป แต่ถ้าหากเป็นคำสั่งอื่น จะทำการเรียกใช้ execvp เพื่อทำตามคำสั่งนั้นเมื่อทำงานเสร็จ จะเข้าสู่ลูปใหม่และทำงานซ้ำต่อไป