시스템 프로그래밍

9주차 process **12224037 김용빈**

1-1) Try below and explain why the output is "I am ex1" when ex2 runs.

```c
#include <unistd.h>
#include <stdio.h>
void main(){
        char *argv[5];
        argv[0] = "./ex1";
        argv[1] = 0;
        execve(argv[0], argv, 0);
        printf("I am ex2\n");
}
```

```c
#include <unistd.h>
#include <stdio.h>
void main(){
        char *argv[5];
        argv[0] = "./ex1";
        argv[1] = 0;
        int y = execve(argv[0], argv, 0);
        if(y < 0) {
                perror("err:");
                exit(0);
        }
        printf("I am ex2\n");
}
```

```
82102@klay ~
$ vi ex2.c

82102@klay ~
$ gcc -o ex2 ex2.c

82102@klay ~
$ ex2
I am ex1
```

문제에서 주어진 코드대로 ex1.c와 ex2.c를 작성하였고 gcc로 디버그 한 후 ex2를 실행하였다. Ex2에서 execve를 하였기에 ex1으로 프로그램이 로드되어 ex1의 코드인 printf("I am ex1);이 실행된다. 위 첫 번째 코드는 오류 점검 없는 ex2이고 두 번째는 오류를 체크하는 코드이다.

1-2) Run myexec below. Explain the result.

```c
#include <stdio.h>
#include <unistd.h>
void main(){
        char *k[10];
        k[0] = "/bin/ls";
        k[1] = 0;
        execve(k[0], k, 0);
}
```

```
82102@klay ~
$ gcc -o myexec myexec.c

82102@klay ~
$ myexec
''$'\001'
''$'\001\005'
'T'$'\034'
']jo'$'\357\203\270\357\202\223\357\203\251
 cdssetup
 cphw.c
 cphw4
 cphw4.c
'd'$'\004'
 d1
 d2
 d3
 ex0.c
 ex0.exe
 ex1.c
 ex1.exe
 ex2.c
 ex2.exe
```

변환하려는 fname이 "/bin/ls"이다. 커맨드 라인 argument로 아무것도 주어지지 않았기에 ls와 같은 결과를 출력한다.

1-3) Run myexec below. Explain the result.

```c
#include <stdio.h>
#include <unistd.h>
void main(){
        char *k[10];
        k[0] = "/bin/ls";
        k[1] = "-l";
        k[2] = 0;
        execve(k[0], k, 0);
}
```

```
82102@klay ~
$ gcc -o myexec myexec.c

82102@klay ~
$ myexec
total 2280
-rwxr-xr-x+ 1 82102 82102      0 Mar 29 16:43 ''$'\001'
-rwxr-xr-x+ 1 82102 82102      0 Mar 29 16:43 ''$'\001\005'
-rwxr-xr-x+ 1 82102 82102      0 Mar 29 16:16 'T'$'\034'
-rwxr-xr-x+ 1 82102 82102      0 Mar 29 16:43 ']jO'$'\357\203\2
1\027\357\203\251\357\203\251\003'
drwx------+ 1 82102 82102      0 Apr  3  2023  cdssetup
-rwxr-xr-x+ 1 82102 82102    295 Mar 27 09:39  cphw.c
-rwxr-xr-x+ 1 82102 82102  66926 Mar 27 09:46  cphw4
-rwxr-xr-x+ 1 82102 82102    295 Mar 27 09:54  cphw4.c
-rwxr-xr-x+ 1 82102 82102      0 Mar 29 16:43 'd'$'\004'
drwxr-xr-x+ 1 82102 82102      0 Mar  4 09:34  d1
drwxr-xr-x+ 1 82102 82102      0 Mar  6 09:27  d2
drwxr-xr-x+ 1 82102 82102      0 Mar 13 09:23  d3
-rw-r--r--+ 1 82102 82102    165 Apr 15 09:26  ex0.c
-rwxr-xr-x+ 1 82102 82102  66203 Apr 15 09:26  ex0.exe
-rw-r--r--+ 1 82102 82102     58 Apr 29 09:23  ex1.c
-rwxr-xr-x+ 1 82102 82102  65814 Apr 29 09:24  ex1.exe
-rw-r--r--+ 1 82102 82102    209 Apr 29 09:30  ex2.c
-rwxr-xr-x+ 1 82102 82102  66007 Apr 29 09:26  ex2.exe
```

Command line argument argv로 -l를 넣어준다. 따라 ls -l과 같은 결과가 출력된다.

1-4) Run myexec below and explain the result.

```c
#include <stdio.h>
#include <unistd.h>
void main(){
        char *k[10];
        k[0] = "/bin/cat";
        k[1] = "f1";
        k[2] = 0;
        execve(k[0], k, 0);
}
```

```
82102@klay ~
$ vi myexec.c

82102@klay ~
$ gcc -o myexec myexec.c

82102@klay ~
$ myexec
00000000: 4d5a 9000 0300 0000 0400 0000 ffff 0000  MZ..............
00000010: b800 0000 0000 0000 4000 0000 0000 0000  ........@.......
00000020: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000030: 0000 0000 0000 0000 0000 0000 8000 0000  ................
00000040: 0e1f ba0e 00b4 09cd 21b8 014c cd21 5468  ........!..L.!Th
00000050: 6973 2070 726f 6772 616d 2063 616e 6e6f  is program canno
00000060: 7420 6265 2072 756e 2069 6e20 444f 5320  t be run in DOS
00000070: 6d6f 6465 2e0d 0d0a 2400 0000 0000 0000  mode....$.......
00000080: 5045 0000 6486 1200 3553 2866 00cc 0000  PE..d...5S(f....
00000090: cb02 0000 f000 2600 0b02 022a 000a 0000  ......&....*....
000000a0: 0020 0000 0002 0000 0010 0000 0010 0000  . ..............
000000b0: 0000 4000 0100 0000 0010 0000 0002 0000  ..@.............
000000c0: 0400 0000 0000 0000 0500 0200 0000 0000  ................
000000d0: 00a0 0100 0006 0000 53fc 0100 0300 0080  ........S.......
000000e0: 0000 2000 0000 0000 0010 0000 0000 0000  .. ............
```

변환하려는 프로그램 이름이 /bin/cat이고 argv로 f1을 넣어주었기에 cat f1과 같은 결과가 출력된다. 실행결과를 보면 f1에 wav파일 저장되어 있기에 cat으로 f1의 내용이 출력된다.

2) Change myexec such that it execs to “/bin/ls -l -a”. Note "l" is small L, not number 1.

```c
#include <stdio.h>
#include <unistd.h>
void main(){
        char *k[10];
        k[0] = "/bin/ls";
        k[1] = "-l";
        k[2] = "-a";
        k[3] = 0;
        execve(k[0], k, 0);
}
```

```
82102@klay ~
$ gcc -o myexec myexec.c

82102@klay ~
$ myexec
total 2327
-rwxr-xr-x+ 1 82102          82102       0 Mar 29 16:43 ''$'\001'
-rwxr-xr-x+ 1 82102          82102       0 Mar 29 16:43 ''$'\001\005'
drwxrwx---+ 1 Administrators 82102       0 Apr 29 09:38  .
drwxrwx---+ 1 82102          SYSTEM      0 Apr 26 16:56  ..
-rw-------+ 1 82102          82102    5898 Apr 24 10:30  .bash_history
-rw-r--r--+ 1 82102          82102      47 Apr  8 10:01  .bash_profile
-rw-r--r--+ 1 82102          82102      20 Apr  8 10:01  .bashrc
-rw-------+ 1 82102          82102      20 Mar  6 10:00  .lesshst
-rw-------+ 1 82102          82102   10420 Apr 29 09:38  .viminfo
-rwxr-xr-x+ 1 82102          82102       0 Mar 29 16:16 'T'$'\034'
-rwxr-xr-x+ 1 82102          82102       0 Mar 29 16:43 ']jo'$'\357\203\270\
357\203\251\027\357\203\251\357\203\251\003'
drwx------+ 1 82102          82102       0 Apr  3  2023  cdssetup
-rwxr-xr-x+ 1 82102          82102     295 Mar 27 09:39  cphw.c
-rwxr-xr-x+ 1 82102          82102   66926 Mar 27 09:46  cphw4
-rwxr-xr-x+ 1 82102          82102     295 Mar 27 09:54  cphw4.c
-rwxr-xr-x+ 1 82102          82102       0 Mar 29 16:43 'd'$'\004'
-rw-r--r--+ 1 82102          82102     405 Apr 14 21:33  mycp.c
-rwxr-xr-x+ 1 82102          82102   66581 Apr 14 21:32  mycp.exe
-rw-r--r--+ 1 82102          82102     306 Apr 14 00:10  myecho.c
-rwxr-xr-x+ 1 82102          82102   66203 Apr 14 00:09  myecho.exe
-rw-r--r--+ 1 82102          82102     149 Apr 29 09:38  myexec.c
-rwxr-xr-x+ 1 82102          82102   65816 Apr 29 09:38  myexec.exe
-rw-r--r--+ 1 82102          82102     360 Apr 14 22:02  myxxd.c
-rwxr-xr-x+ 1 82102          82102   66582 Apr 14 22:02  myxxd.exe
-rwxr-xr-x+ 1 82102          82102     295 Mar 27 10:10  newhw4.c
-rw-r--r--+ 1 82102          82102    1335 Apr  6 21:50  nontext.c
-rwxr-xr-x+ 1 82102          82102   68144 Apr  6 21:50  nontext.exe
-rwxr-xr-x+ 1 82102          82102   69328 Mar 22 13:35  str1.
-rw-r--r--+ 1 82102          82102    2107 Mar 23 14:17  str1.c
-rwxr-xr-x+ 1 82102          82102   69912 Mar 23 14:18  str1.exe
-rw-r--r--+ 1 82102          82102     202 Mar 20 18:51  str2.c
-rwxr-xr-x+ 1 82102          82102   65816 Mar 20 18:51  str2.exe
-rw-r--r--+ 1 82102          82102       0 Apr  3 09:24  sw2-wav.txt
-rwx------+ 1 82102          82102   30268 Apr  3 10:08  sw2.wav
-rwx------+ 1 82102          82102   60492 Apr  6 21:18  sw3.wav
-rw-r--r--+ 1 82102          82102       0 Apr 24 09:29  sw4.wav
-rwx------+ 1 82102          82102   30268 Apr 24 10:09  swvader03.wav
-rw-r--r--+ 1 82102          82102  257104 Apr  6 21:21  x
-rw-r--r--+ 1 82102          82102      54 Mar 13 09:16  y.c
```

/bin/ls로 프로그램 이름을 넣어주고 -l과 -a를 argv에 넣었다 이때 -a로 한 argument가 추가되었으므로 k의 마지막 인덱스인 k[3] = 0으로 할당해야 한다.

3) Change myexec such that it execs to “/bin/cp f1 f2”.

```c
#include <stdio.h>
#include <unistd.h>
void main(){
        char *k[10];
        k[0] = "/bin/cp";
        k[1] = "f1";
        k[2] = "f2";
        k[3] = 0;
        execve(k[0], k, 0);
}
```

```
82102@klay ~
$ gcc -o myexec myexec.c

82102@klay ~
$ cat f1
I am f1


82102@klay ~
$ myexec

82102@klay ~
$ cat f2
I am f1
```

변환하려는 프로그램 이름을 /bin/cp로 하였고 argv에 f1, f2를 넣어주었다. 따라 cp f1 f2를 실행한 결과와 같은 결과가 출력된다. F1에 I am f1이 저장되어 있었고 myexec 실행 후 cat f2를 하면 f1의 내용이 f2에 복사됨을 확인할 수 있다.

4) Change myexec such that it runs "/bin/ls -l" and prints "job done" after "/bin/ls -l" is finished

```c
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <sys/wait.h>
void main(){
        char *k[10];
        k[0] = "/bin/ls";
        k[1] = "-l";
        k[2] = 0;
        if(fork() == 0){
                execve(k[0], k, 0);
                exit(0);
        }
        else {
                wait(NULL);
                printf("job done\n");
        }
}
```

```
82102@klay ~
$ myexec
total 1997
-rwxr-xr-x+ 1 82102 82102      0 Mar 29 16:43 ''$'\001'
-rwxr-xr-x+ 1 82102 82102      0 Mar 29 16:43 ''$'\001\005'
-rwxr-xr-x+ 1 82102 82102      0 Mar 29 16:16 'T'$'\034'
-rwxr-xr-x+ 1 82102 82102      0 Mar 29 16:43 ']jO'$'\357\2
1\027\357\203\251\357\203\251\003'
drwx------+ 1 82102 82102      0 Apr  3  2023  cdssetup
-rwxr-xr-x+ 1 82102 82102    295 Mar 27 09:39  cphw.c
-rwxr-xr-x+ 1 82102 82102  66926 Mar 27 09:46  cphw4
-rwxr-xr-x+ 1 82102 82102    295 Mar 27 09:54  cphw4.c
-rwxr-xr-x+ 1 82102 82102      0 Mar 29 16:43 'd'$'\004'
drwxr-xr-x+ 1 82102 82102      0 Mar  4 09:34  d1
-rw-r--r--+ 1 82102 82102    380 Apr 14 22:02  myxxd.c
-rwxr-xr-x+ 1 82102 82102  66582 Apr 14 22:02  myxxd.exe
-rwxr-xr-x+ 1 82102 82102    295 Mar 27 10:10  newhw4.c
-rw-r--r--+ 1 82102 82102   1335 Apr  6 21:50  nontext.c
-rwxr-xr-x+ 1 82102 82102  68144 Apr  6 21:50  nontext.exe
-rwxr-xr-x+ 1 82102 82102  69328 Mar 22 13:35  str1.
-rw-r--r--+ 1 82102 82102   2107 Mar 23 14:17  str1.c
-rwxr-xr-x+ 1 82102 82102  69912 Mar 23 14:18  str1.exe
-rw-r--r--+ 1 82102 82102    202 Mar 20 18:51  str2.c
-rwxr-xr-x+ 1 82102 82102  65816 Mar 20 18:51  str2.exe
-rw-r--r--+ 1 82102 82102      0 Apr  3 09:24  sw2-wav.txt
-rwx------+ 1 82102 82102  30268 Apr  3 10:08  sw2.wav
-rwx------+ 1 82102 82102  60492 Apr  6 21:18  sw3.wav
-rw-r--r--+ 1 82102 82102      0 Apr 24 09:29  sw4.wav
-rwx------+ 1 82102 82102  30268 Apr 24 10:09  swvader03.wav
-rw-r--r--+ 1 82102 82102 257104 Apr  6 21:21  x
-rw-r--r--+ 1 82102 82102     54 Mar 13 09:16  y.c
job done
```

Execve를 하면 현재 프로그램이 지워지므로 밑의 코드가 실행되지 않는다. 따라 새로운 프로세스를 만들어 주어야 한다. Fork()로 자식 프로세스에서 execve 명령어를 실행 한 후 자식 프로세스가 종료되면 부모 프로세스에서 wait()으로 기다리다가 "job done"을 출력하도록 코드를 작성하였다. Wait(NULL)로 자식 프로세스가 종료됨을 기다리는데 <sys/wait.h> 헤더파일을 사용하였다.

5) Change myexec such that it reads a command and execs to the given command. Your code should be able to exec any command. Read the command line with gets() or fgets() and use strtok() to extract all arguments and perform exec to run the command. Remember you have to remove the last character('₩n') if you use fgets.

```c
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <string.h>
void main(){
        char *argv[25];
        char buf[50];
        printf("command > ");
        fgets(buf, 50, stdin);
        buf[strlen(buf) - 1] = '\0' ;

        int index = 0;
        argv[index] = strtok(buf, " ");
        while(argv[index] != NULL){
                index = index + 1;
                argv[index] = strtok(NULL, " ");
        }
        execve(argv[0], argv, NULL);
}
```

```
82102@klay ~
$ gcc -o myexec myexec.c

82102@klay ~
$ myexec
command > /bin/cat f1
I am f1


82102@klay ~
$ myexec
command > /bin/cp f1 f2

82102@klay ~
$ cat f2
I am f1
```

Argv를 위한 char pointer array를 선언하였고 입력하는 문자열을 저장하는 buf를 선언하였다. Fgets()로 문자열을 입력 받았고 buf의 마지막 문자를 0으로 할당하여 저장된 데이터가 문자열이 되도록 하였다. 다음으로 index = 0 부터 strtok()를 사용하여 argv[index]에 strtok로 추출한 문자열을 저장하도록 하였다. Argv[index]가 NULL이 될 때까지 while 루프를 돌리며 argument들을 추출한다. 다음으로 execve를 하여 입력한 프로그램 즉 argv[0]에 저장된 fname을 갖는 프로그램을 로드하여 실행한다.

결과를 보면 문자열을 입력하고 argv를 추출하여 해당 프로그램이 실행됨을 확인할 수 있다.

6) Same as 5), but myexec will repeat the process forever until you stop the program with "ctrl-c". Also display the result of strtok as shown below.

```c
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <sys/wait.h>
int main(){
//      char dirname[256];
//      getcwd(dirname, 256);
        for(int i =0; ; i++){

                char *argv[100];
                char buf[100];
//              printf("[%s]$ ", dirname);
                printf("command > ");
                fgets(buf, 100, stdin);
                buf[strlen(buf) - 1] = '\0' ;

//              if(strcmp(buf, "exit") == 0) break;

                int index = 0;
                argv[index] = strtok(buf, " ");
                while(argv[index] != NULL){
                        index += 1;
                        argv[index] = strtok(NULL, " ");
                }
                argv[index] = NULL;

                printf("argc: %d ", index);
                for(int j=0; j< index; j++){
                        printf("argv[%d]: %s ", j, argv[j]);
                }
                printf("\n");

                if(fork() == 0){
                        execve(argv[0], argv, NULL);
                }
                else {
                        wait(NULL);
                }

        }
        return 0;
}
```

```
82102@klay ~
$ g++ -o myexec myexec.c

82102@klay ~
$ myexec
command > /bin/cat f1
argc: 2 argv[0]: /bin/cat argv[1]: f1
I am f1

command > /bin/cp f1 f2
argc: 3 argv[0]: /bin/cp argv[1]: f1 argv[2]: f2
command > /bin/cat f2
argc: 2 argv[0]: /bin/cat argv[1]: f2
I am f1

command >

82102@klay ~
$
```

5번과 같이 계속 command를 입력 받고 execve를 수행하여 원하는 프로그램을 실행하는 코드를 작성한다. 6번은 5번과 다르게 argc의 개수, argv[]에 저장된 argument를 화면에 출력해야 한다. 5번에서 해결한 것과 다르게 먼저 while문을 통해 strtok()으로 입력된 문자열에서 argument들을 argv[]에 저장한다. 다음으로 argc는 index 값으로, argument들은 argv[]에 저장된 문자들을 출력하

여 화면에 나타낸다. 그 후에 if else문에서 fork()로 자식 프로세스이면 execve를 하여 입력 받은 프로그램을 실행하도록 하고 부모 프로세스이면 자식 프로세스의 종료를 기다린다.

결과를 보면 /bin/cat f1이 입력되고 argc는 argument 개수가 2개 이므로 2가 출력되고 각 argv[] 배열에 저장된 argument가 출력된다.

7) Same as 6), but change the prompt to the current location and '$' as follows. You may need "getcwd". Also add code to detect "exit" command and stop the program when the user types "exit".

```c
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <sys/wait.h>
int main(){
        char dirname[256];
        getcwd(dirname, 256);
        for(int i =0; ; i++){

                char *argv[100];
                char buf[100];
                printf("[%s]$ ", dirname);
                fgets(buf, 100, stdin);
                buf[strlen(buf) - 1] = '\0' ;

                if(strcmp(buf, "exit") == 0) break;

                int index = 0;
                argv[index] = strtok(buf, " ");
                while(argv[index] != NULL){
                        index += 1;
                        argv[index] = strtok(NULL, " ");
                }
                argv[index] = NULL;

                printf("argc: %d ", index);
                for(int j=0; j< index; j++){
                        printf("argv[%d]: %s ", j, argv[j]);
                }
                printf("\n");

                if(fork() == 0){
                        execve(argv[0], argv, NULL);
                }
                else {
                        wait(NULL);
                }

        }
        return 0;
}
```

```
82102@klay ~
$ g++ -o myexec myexec.c

82102@klay ~
$ myexec
[/cygdrive/c/Users/82102/AppData/Roaming/SPB_Data]$ /bin/cat f1
argc: 2 argv[0]: /bin/cat argv[1]: f1
I am f1

[/cygdrive/c/Users/82102/AppData/Roaming/SPB_Data]$ /bin/cp f1 f2
argc: 3 argv[0]: /bin/cp argv[1]: f1 argv[2]: f2
[/cygdrive/c/Users/82102/AppData/Roaming/SPB_Data]$ /bin/cat f2
argc: 2 argv[0]: /bin/cat argv[1]: f2
I am f1

[/cygdrive/c/Users/82102/AppData/Roaming/SPB_Data]$ exit
```

6번과 매우 비슷한 방식으로 코드를 작성하였다. 추가한 부분은 directory 이름을 가져오는 getcwd함수를 사용한 것인데, 현재 디렉토리 이름을 dirname 저장하였다. 5,6번에서는 printf("command > ")를 하였는데 이 코드에서는 printf("[%s]$ ", dirname)을 하여 현재 디렉토리

이름이 출력되도록 하였다.

출력을 보면 현재 디렉토리 위치가 출력되고 뒤에 입력할 수 있게 되어있다. argument들을 출력하고 실행하는 동작은 5번 6번에서 구현한 방식과 동일하다.