

시스템 프로그래밍 실습

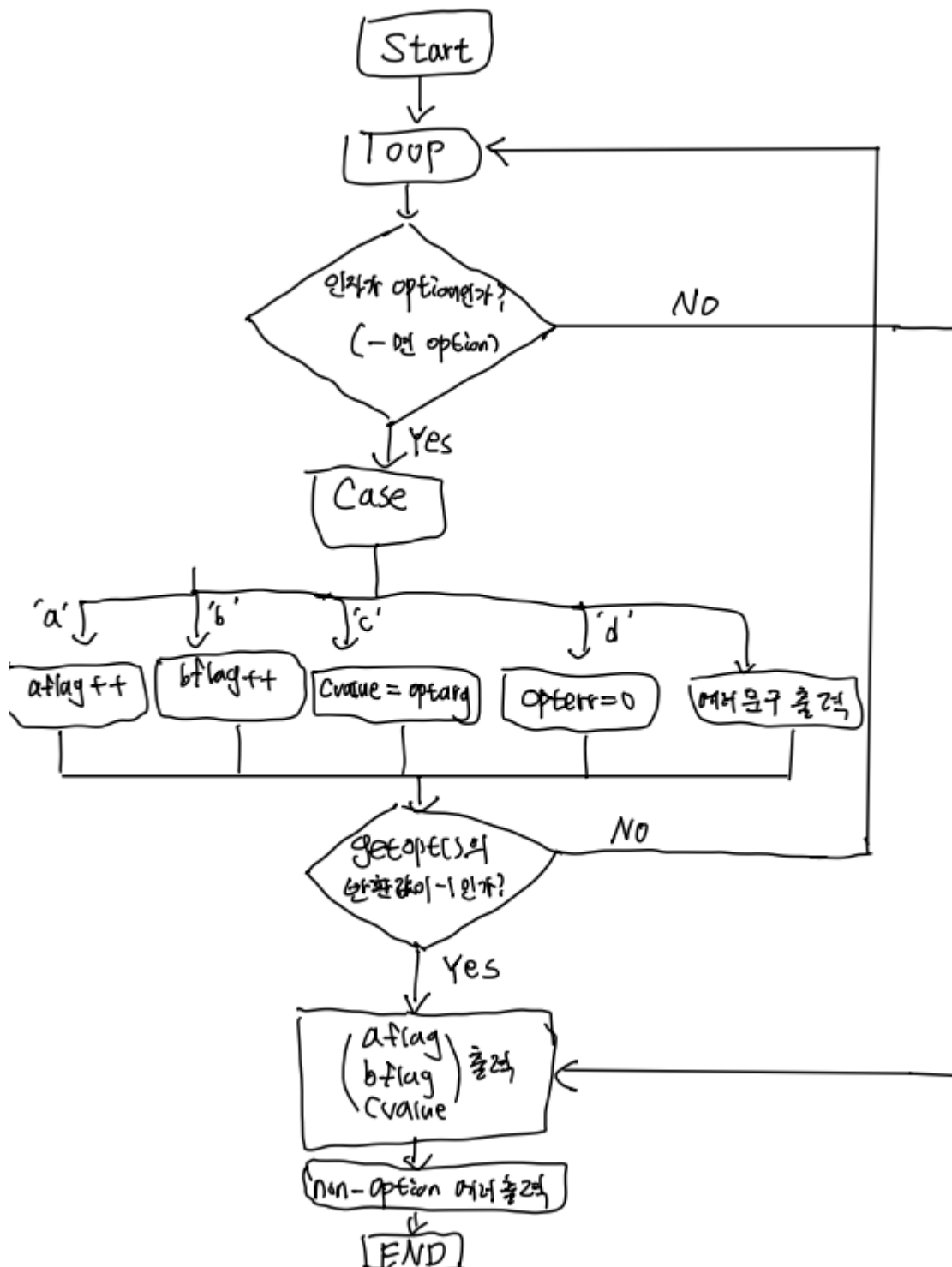
# [Assignment 1-1]

Class : D  
Professor : 최상호 교수님  
Student ID : 2020202092  
Name : 강현민

# Introduction

이번 과제는 FTP 서버를 직접 구현하는 것을 최종 목표로 하는 프로젝트의 첫 번째 과정인 FTP command 를 구현하는 과정으로, 그 중에서도 명령어의 option(-a, -b 등) 및 인자를 getopt() 함수를 이용하여 파싱하는 것을 목적으로 한다. 이 때 옵션 파싱을 위해 getopt 함수는 반복적으로 호출되며, 반복문 내부에서 switch 문의 case 를 이용하여 입력 받은 option 에 대한 적절한 동작을 수행한다. getopt() 함수가 -1 을 반환하는 경우 더 이상 옵션이 없기 때문에 파싱 과정은 종료되고, 0 을 반환하는 경우 계속해서 반복문을 통해 파싱을 수행하게 된다.

## Flow chart



처음 시작하면 main 함수가 호출되고, while 문의 조건으로 getopt 함수가 호출된다. 그리고 while 문 안에서는 case 를 통해 입력받은 option 을 파싱하는데, 옵션이 a 일 경우 aflag 를 1 증가시킨 후 break 하고, b 일 경우 bflag 를 1 증가시키고 break 하고, c 일 경우

c와 함께 입력된 string 이 저장된 optarg를 cvalue에 저장한 후 break한다. 그리고 d일 경우 opterr 시그널을 0으로 set한 후 break하고, ? 즉 정의되지 않은 option일 경우 에러 문구를 출력한 후 break한다. 리턴값이 -1이 아니면 반복하고, 리턴값이 -1이면 반복을 끝낸다. 그 후 aflag, bflag, cvalue를 출력한 후 non-option이 있다면 에러 메시지를 모두 출력한 후 종료한다.

# Pseudo code

Int main (int argc, char\*\* argv)

{

    Declare and initialize int aflag = 0, bflag = 0;

    Declare and initialize \*cvalue = NULL;

    Declare and initialize int opterr = 0;

    While ( return value of getopt() is not -1)

    {

        If option is encountered:

        {

            Case 'a':

                Increment aflag;

                break;

            case 'b':

                increment bflag;

                break;

            case 'c':

                set cvalue as inputted arguments (optarg);

                break;

            case '?'

                print "Unknown option character";

                break;

        }

}

output value of aflag, bflag and string of cvalue;

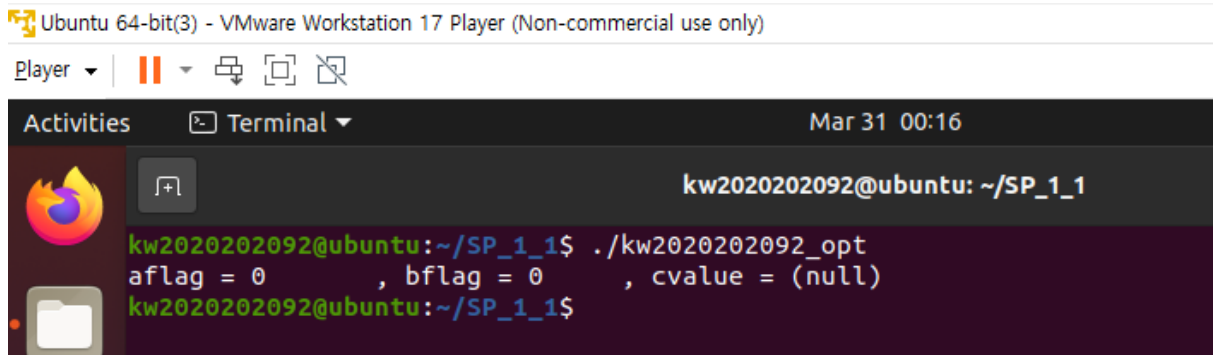
Output all non-option arguments

Exit program

}

## 결과화면

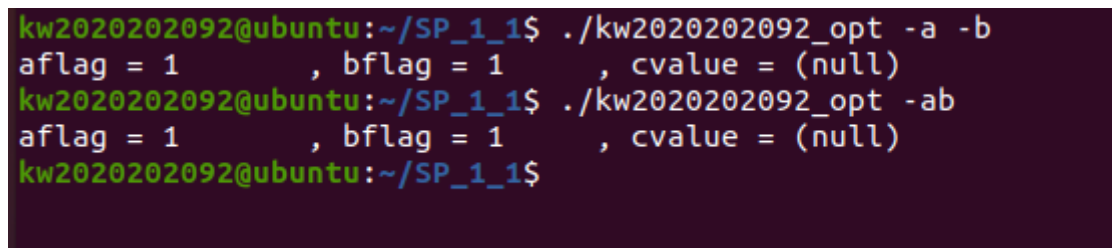
- 1) 옵션이 없는 경우



```
Ubuntu 64-bit(3) - VMware Workstation 17 Player (Non-commercial use only)
Player | [Pause] [Full Screen] [Close]
Activities Terminal Mar 31 00:16
kw2020202092@ubuntu: ~/SP_1_1
kw2020202092@ubuntu:~/SP_1_1$ ./kw2020202092_opt
aflag = 0 , bflag = 0 , cvalue = (null)
kw2020202092@ubuntu:~/SP_1_1$
```

옵션이 없는 경우에는 위와 같이 aflag, bflag 와 cvalue 에 아무것도 저장되지 않으므로 위와 같은 결과가 출력되는 것을 확인할 수 있다.

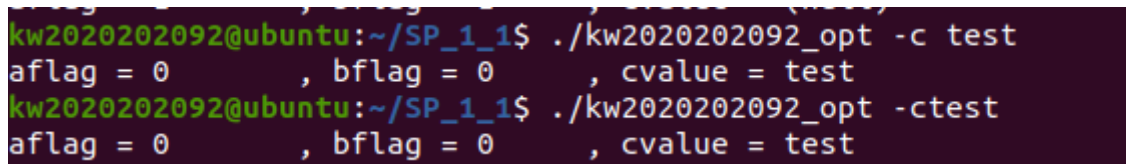
- 2) 옵션으로 a 와 b 가 입력된 경우



```
kw2020202092@ubuntu:~/SP_1_1$ ./kw2020202092_opt -a -b
aflag = 1 , bflag = 1 , cvalue = (null)
kw2020202092@ubuntu:~/SP_1_1$ ./kw2020202092_opt -ab
aflag = 1 , bflag = 1 , cvalue = (null)
kw2020202092@ubuntu:~/SP_1_1$
```

위와 같이 옵션 a 와 b 가 따로 입력되는 경우에 aflag 와 bflag 가 1 증가하여 출력되는 것을 확인할 수 있고, a 와 b 가 -ab 와 같이 한번에 입력되는 경우에도 같은 결과가 출력되는 것을 확인할 수 있다.

- 3) 옵션으로 c 와 string 이 입력된 경우



```
kw2020202092@ubuntu:~/SP_1_1$ ./kw2020202092_opt -c test
aflag = 0 , bflag = 0 , cvalue = test
kw2020202092@ubuntu:~/SP_1_1$ ./kw2020202092_opt -ctest
aflag = 0 , bflag = 0 , cvalue = test
```

위와 같이 -c "test"형식으로 입력된 경우에 cvalue 로 string test 가 저장되는 것을 확인할 수 있다. 그리고 띄어쓰기 없이 -ctest 로 입력되는 경우에도 마찬가지로 cvalue 에 string test 가 저장되는 것을 확인할 수 있다.

4) option 으로 string 이 입력된 경우

```
kw2020202092@ubuntu:~/SP_1_1$ ./kw2020202092_opt test
aflag = 0      , bflag = 0      , cvalue = (null)
Non-option argument test
kw2020202092@ubuntu:~/SP_1_1$ ./kw2020202092_opt -a test
aflag = 1      , bflag = 0      , cvalue = (null)
Non-option argument test
kw2020202092@ubuntu:~/SP_1_1$ ./kw2020202092_opt -b test
aflag = 0      , bflag = 1      , cvalue = (null)
Non-option argument test
kw2020202092@ubuntu:~/SP_1_1$ ./kw2020202092_opt -c test test2
aflag = 0      , bflag = 0      , cvalue = test
Non-option argument test2
kw2020202092@ubuntu:~/SP_1_1$
```

위와 같이 명령어의 인자로 바로 string 이 입력된 경우 non-option argument 에러메세지가 출력되는 것을 확인할 수 있다. 또한, 옵션으로 a 혹은 b 를 입력한 후 string 이 입력된 경우 그 앞의 option a 혹은 b 는 정상적으로 동작하여 aflag 와 bflag 가 1 증가하지만, 그 뒤의 string 은 option 이 아니기 때문에 non-option 에러 문구가 출력되는 것을 확인할 수 있다. 또한, 옵션으로 c 가 입력된 경우 그 뒤의 string 은 cvalue 로 잘 저장되지만, 그 뒤의 string 은 non-option 에러 문구가 출력되는 것을 확인할 수 있다.

```
kw2020202092@ubuntu:~/SP_1_1$ ./kw2020202092_opt -a -
aflag = 1      , bflag = 0      , cvalue = (null)
Non-option argument -
kw2020202092@ubuntu:~/SP_1_1$
```

또한 위와 같이 옵션으로 a 가 입력되고 abcd 없이 -만 입력된 경우, non-option 으로 분류되어 에러 문구가 출력되는 것을 확인할 수 있다.



5) 같은 옵션이 2 개 이상 입력되는 경우

```
kw2020202092@ubuntu:~/SP_1_1$ ./kw2020202092_opt -aa
aflag = 2      , bflag = 0      , cvalue = (null)
kw2020202092@ubuntu:~/SP_1_1$ ./kw2020202092_opt -bb
aflag = 0      , bflag = 2      , cvalue = (null)
kw2020202092@ubuntu:~/SP_1_1$ ./kw2020202092_opt -aa -bb
aflag = 2      , bflag = 2      , cvalue = (null)
kw2020202092@ubuntu:~/SP_1_1$ ./kw2020202092_opt -aabb
aflag = 2      , bflag = 2      , cvalue = (null)
kw2020202092@ubuntu:~/SP_1_1$
```

위와 같이 같은 옵션이 두개 이상 입력되는 경우, 정상적으로 aflag 와 bflag 가 2 증가하여 출력되는 것을 확인할 수 있으며, -aabb 와 같이 동시에 입력하는 경우에도 의도대로 잘 출력되는 것을 확인할 수 있다.

6) option 으로 d 가 입력된 경우

```
kw2020202092@ubuntu:~/SP_1_1$ ./kw2020202092_opt -d
aflag = 0      , bflag = 0      , cvalue = (null)
kw2020202092@ubuntu:~/SP_1_1$ ./kw2020202092_opt -d -a -b
aflag = 1      , bflag = 1      , cvalue = (null)
kw2020202092@ubuntu:~/SP_1_1$
```

Getopt 함수의 내부 변수인 opterr 에 0 을 저장하는 case 인 옵션 d 가 입력되는 경우 출력상으로는 아무 값도 변하지 않는 것을 확인할 수 있다.

7) Non-option 이 여러 개인 경우

```
kw2020202092@ubuntu:~/SP_1_1$ ./kw2020202092_opt test test test test
aflag = 0      , bflag = 0      , cvalue = (null)
Non-option argument test
Non-option argument test
Non-option argument test
Non-option argument test
kw2020202092@ubuntu:~/SP_1_1$ ./kw2020202092_opt -a test test test
aflag = 1      , bflag = 0      , cvalue = (null)
Non-option argument test
Non-option argument test
Non-option argument test
kw2020202092@ubuntu:~/SP_1_1$
```

위와 같이 non-option 이 여러 개 입력되는 경우에 모든 non-option 에 대한 여러 문구를 출력하는 것을 확인할 수 있다.

## 고찰

이번 과제를 통해 unistd 헤더파일의 getopt 함수를 통해 입력 받은 인자를 분석하여 옵션 처리를 할 수 있다는 점을 학습하고 이를 응용하여 FTP command 의 option 을 파싱하여 option flag 를 출력하는 코드를 구현할 수 있었다. 또한, 지난 실습 때 학습한 Makefile 및 리눅스의 명령어를 다시 한번 사용하면서 응용하면서 리눅스 프로그래밍에 익숙해질 수 있었다.

## Reference