CS5243 Advanced UNIX Programming
Assignment 11 (4 pts)
Group 4

(1) Screenshot of the code

```c
int main(int argc, char**argv){

    // Call daemonize() function
    daemonize("assignment11");

    // Call getlogin() function
    char *str = getlogin();
    printf("%s\n", str);

    // Open file
    FILE *f;
    /*  The path should be the absolute path */
    /*  Since the directory is changed to the root */
    // f = fopen("/home/ccw/Unix/Project/assignment11/assignment11.txt", "w");
    f = fopen("/home/freebsd/Advanced-UNIX-Programming_Student/assignment11/
    assignment11.txt", "w");

    // Write login name in the file
    fprintf(f, "Login name: %s\n", str);

    // Close file
    fclose(f);

    return 0;
}
```

(2) Screenshot of the result

```
freebsd@generic:~/Advanced-UNIX-Programming_Student/assignment11 % ./assignment11
freebsd@generic:~/Advanced-UNIX-Programming_Student/assignment11 % cat assignment11.txt
Login name: freebsd
freebsd@generic:~/Advanced-UNIX-Programming_Student/assignment11 %
```

(3) Explanation:

```c
void daemonize(const char *cmd)
{
    int i, fd0, fd1, fd2;
    pid_t pid;
    struct rlimit rl = {};
    struct sigaction sa = {};

    /*
     * Clear file creation mask.
     */
    umask(0);

    /*
     * Get maximum number of file descriptors.
     */
    if (getrlimit(RLIMIT_NOFILE, &rl) < 0)
        err_quit("%s: can't get file limit", cmd);
```

將umask設為0, 讓daemon process 如果要創建文件時是group-read, group-write.
getrlimit確認能取得file descriptor

```c
/*
 * Become a session leader to lose controlling TTY.
 */
if ((pid = fork()) < 0)
    err_quit("%s: can't fork", cmd);
else if (pid != 0) /* parent */
    exit(0);
setsid();

/*
 * Ensure future opens won't allocate controlling
 TTYs.
 */
sa.sa_handler = SIG_IGN;
sigemptyset(&sa.sa_mask);

sa.sa_flags = 0;
if (sigaction(SIGHUP, &sa, NULL) < 0)
    err_quit("%s: can't ignore SIGHUP", cmd);
if ((pid = fork()) < 0)
    err_quit("%s: can't fork", cmd);
else if (pid != 0) /* parent */
    exit(0);
```

呼叫 fork()和setsid()，讓daemon變成 new session leader並中斷和terminal的連線
並屏蔽 SIGNHUP

```
/*
 * Change the current working directory to the
 root so
 * we won't prevent file systems from being
 unmounted.
 */
if (chdir("/") < 0)
    err_quit("%s: can't change directory to /",
    cmd);
/*
 * Close all open file descriptors.
 */
if (rl.rlim_max == RLIM_INFINITY)
    rl.rlim_max = 1024;
for (i = 0; i < rl.rlim_max; i++)
    close(i);
```

將路徑改到root 底下
關閉其他不需要的file descriptor

```c
/*
 * Attach file descriptors 0, 1, and 2 to /dev/
 null.
 */
fd0 = open("/dev/null", O_RDWR);
fd1 = dup(0);
fd2 = dup(0);

/*
 * Initialize the log file.
 */
openlog(cmd, LOG_CONS, LOG_DAEMON);
if (fd0 != 0 || fd1 != 1 || fd2 != 2)
{
    syslog(LOG_ERR, "unexpected file descriptors
    %d %d %d", fd0, fd1, fd2);
    exit(1);
}
```

保留file descriptor 0, 1, 2並改到/dev/null下，如果程式讀取寫入時會沒有影響

openlog() 打開log file, 讓daemon能寫入log

(4) Discussion:

process變成daemon後，會在背景繼續執行，並把log寫入到syslog裡，在freebsd裡 log file在/var/log/messages，而且會把執行路徑改到root底下，所以最後開檔案寫入login name時，文件的路徑要是絕對路徑，不然會發生segmentaion fault