BIT

Bahir Dar University institute of Technology

Operating System and System Programming Individual second assignment

On System Call

Prepared by: Tadele Workie Mihretu.

ID: 1308433

```
int getdents(unsigned int fd, struct linux_dirent *dirp,
unsigned int count);
```

Submitted to Instructor Wondimu Baye.

Submission Date: JULY 17, 2014 E.C

CONTENTS

Introduction....3

What / why / how, getdents system call? 3

Briefly describe about the list of parameters.....4

LIst of flags and their implementation 5

Source code implementation using O_RDONLY flag....6

The output of the above source code using the O_RDONLY flag....8

The source code using O_DIRECTORY flag....8

The output of the above source code using O_DIRECTORY 10

Source code implementation by using both flags at the same time. 10

The output of the above source code: 12

Reference: 12

INTRODUCTION

System call is a mechanism that provides the interface between a process and the operating system. It is a programmatic method in which a computer program requests a service from the kernel of the OS.System call offers the services of the operating system to the user programs via API (Application Programming Interface). System calls are the only entry points for the kernel system. A system call **connects to the operating system's kernel, which executes in kernel space**. When an application creates a system call, it must first obtain permission from the kernel. It achieves this using an interrupt request, which pauses the current process and transfers control to the kernel.

Syntax of getdents () system call.

```
#include <sys/syscall.h>       /* Definition of SYS_* constants *
#include <unistd.h>
```

```
int getdents(unsigned int fd, struct linux_dirent *dirp,
unsigned int count);
```

WHAT / WHY / HOW, GETDENTS SYSTEM CALL?

The system call **getdents** () reads several *linux_dirent* structures from the directory referred to by the open file descriptor *fd* into the buffer pointed to by *dirp*. The argument *count* specifies the size of that buffer.
The getdents() function attempts to read nbyte bytes from the directory associated with the file descriptor fildes and to format them as file system independent directory entries in the buffer pointed to by buffer. Since the file system independent directory entries are of variable lengths, in most cases the actual number of bytes returned will be less than nbyte. The file system independent directory entry is specified by the dirent structure.
struct linux dirent will be returned by getdents. It will do this for any underlying file system type. Because the "on disk" format may be completely different, known only to the given file system driver, a simple user space read call may fail. To put it another way, getdents can convert from the native format to fill the linux dirent.
Directory entries in UNIX can refer to files, but also to directories, named pipes and devices. The character and block device entries are the interface to the different drivers in the kernel. They contain a number to identify the driver, and another number to identify different devices handled by the same driver.
Newer systems also support file system sockets, but the whole socket concept is not part of the original UNIX design. Sockets are not files, and "everything is a file" refers to the original UNIX design. But once sockets are set up, they support some operations that also work on files.

RETURN VALUE

On success, the number of bytes read is returned. On end of direc-tory, 0 is returned.
On error, -1 is returned, and errno is set ap-propriately.

ERRORS

EBADF	Invalid file descriptor fd.

EFAULT Argument points outside the calling process's address space.

EINVAL Result buffer is too small.

ENOENT No such directory.

ENOTDIR File descriptor does not refer to a directory.

BRIEFLY DESCRIBE ABOUT THE LIST OF PARAMETERS.

There getdents () has three parameters with during this implementation.

- ***fd***
- **count**
- **struct linux_dirent *dirp**

fd :- file descriptor it is used to access the file.

Count: - it is used to describe the size of buffer

The linux_dirent structure : is declared as follows:

```
struct linux_dirent {

        unsigned long  d_ino;
        unsigned long  d_off;
        unsigned short d_reclen;
        char           d_name[];
        char           pad;
        char           d_type;
     };
```

d_ino *is an inode number.*
d_off *is the distance from the start of the directory to the start of the next linux_dirent.*
d_reclen *is the size of this entire linux_dirent.*
d_name *is a null-terminated filename.*
d_type *is a byte at the end of the structure that indicates the file type.*
It contains one of the following values (defined in <di-rent.h>):

DT_BLK	*This is a block device.*
DT_CHR	*This is a character device.*
DT_DIR	*This is a directory.*

DT_FIFO	*This is a named pipe (FIFO).*
DT_LNK	*This is a symbolic link.*
DT_REG	*This is a regular file.*
DT_SOCK	*This is a UNIX domain socket.*
DT_UNKNOWN	*The file type is unknown.*

There is a two flags also in getdents system call

- **O_RDONLY**

We can use the "O_RDONLY" flag of the C programming language only if we have included the "sys/types.h", "sys/stat.h", and "fcntl.h" header files in our C script. In this simple C program, we have defined an integer type variable "fd" that refers to the file descriptor of the file that we want to open as read-only. Then, we have used the "open()" function of the C programming language and have passed to it the path of the desired file followed by the "O_RDONLY" flag indicating that we want to open the file as read-only. Finally, we have printed a confirmation message on the terminal using the "printf" statement.

- **O_DIRECTORY**

If pathname is not a directory, cause the open to fail. This flag was added in kernel version 2.1.126, to avoid denial-of-service problems if **open()** is called on a FIFO or tape device.

3. List the flags, their purpose with code implementation (give example source code with output)

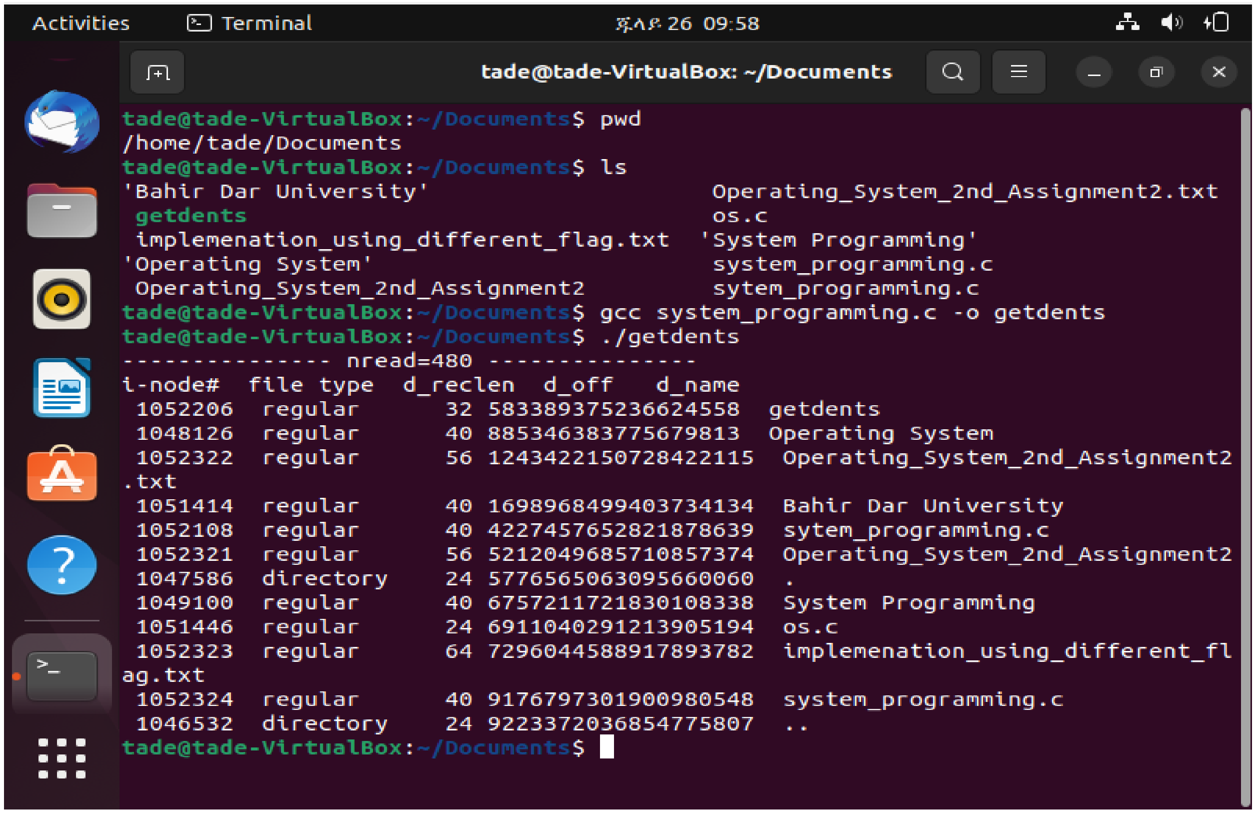
LIST OF FLAGS AND THEIR IMPLEMENTATION

Source code implementation using O_RDONLY flag

```
#define _GNU_SOURCE
#include <dirent.h>
#include <fcntl.h>
#include <stdint.h>
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <sys/stat.h>
#include <sys/syscall.h>
#define handle_error(msg)
              do { perror(msg); exit(EXIT_FAILURE); } while (0)

       struct linux_dirent {
           unsigned long  d_ino;
           off_t          d_off;
           unsigned short d_reclen;
           char           d_name[];
       };

       #define BUF_SIZE 1024

       int
       main(int argc, char *argv[])
       {
           int fd;
           long nread;
           char buf[BUF_SIZE];
           struct linux_dirent *d;
           char d_type;

           fd = open(argc > 1 ? argv[1] : ".", O_RDONLY);
           if (fd == -1)
               handle_error("open");

           for (;;) {
               nread = syscall(SYS_getdents, fd, buf, BUF_SIZE);
               if (nread == -1)
                   handle_error("getdents");

               if (nread == 0)
                   break;

               printf("--------------- nread=%d ---------------\n", nread);
               printf("inode#    file type  d_reclen  d_off   \t\t d_name\n");
```

```
            for (long bpos = 0; bpos < nread;) {
                d = (struct linux_dirent *) (buf + bpos);
                printf("%8ld  ",  d->d_ino);
                d_type = *(buf + bpos + d->d_reclen - 1);
                printf("%-10s ", (d_type == DT_REG) ?  "regular" :
                                 (d_type == DT_DIR) ?  "directory" :
                                 (d_type == DT_FIFO) ? "FIFO" :
                                 (d_type == DT_SOCK) ? "socket" :
                                 (d_type == DT_LNK) ?  "symlink" :
                                 (d_type == DT_BLK) ?  "block dev" :
                                 (d_type == DT_CHR) ?  "char dev" : "???");
                printf("%4d %10jd  %s\n", d->d_reclen,
                        (intmax_t) d->d_off, d->d_name);
                bpos += d->d_reclen;
            }
        }
        exit(EXIT_SUCCESS);
}
```

The output of the above source code using the O_RDONLY flag



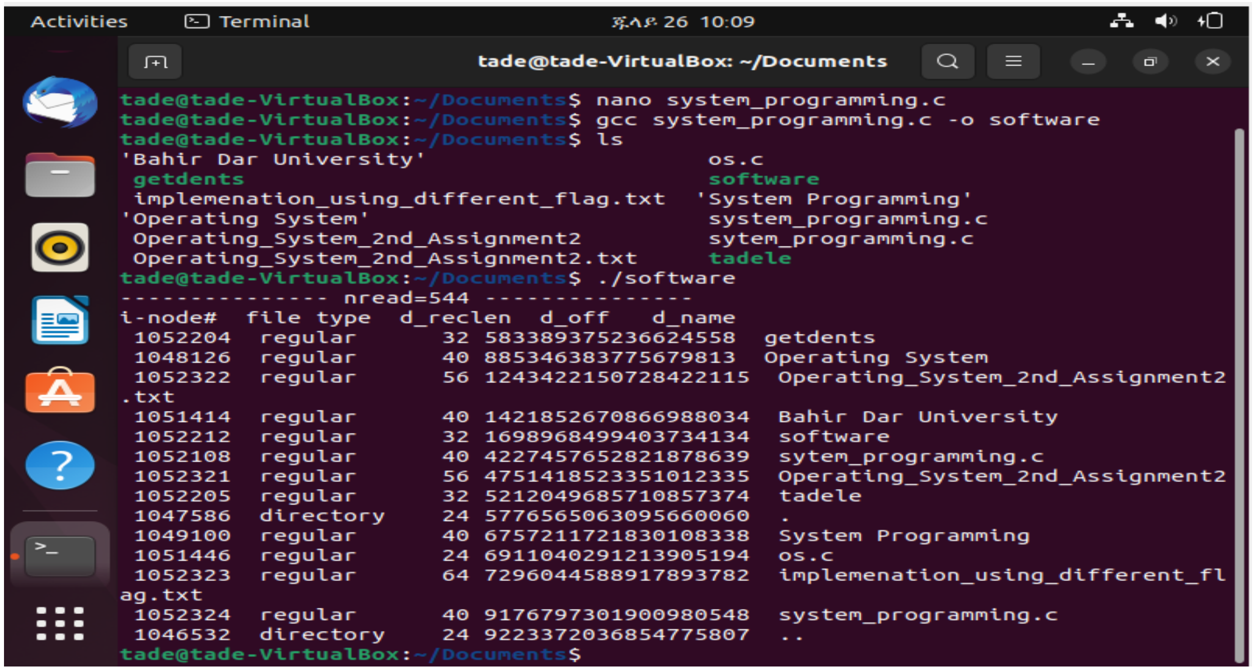
```
tade@tade-VirtualBox:~/Documents$ pwd
/home/tade/Documents
tade@tade-VirtualBox:~/Documents$ ls
'Bahir Dar University'                        Operating_System_2nd_Assignment2.txt
 getdents                                     os.c
 implemenation_using_different_flag.txt      'System Programming'
'Operating System'                            system_programming.c
 Operating_System_2nd_Assignment2             sytem_programming.c
tade@tade-VirtualBox:~/Documents$ gcc system_programming.c -o getdents
tade@tade-VirtualBox:~/Documents$ ./getdents
--------------- nread=480 ---------------
i-node#  file type  d_reclen  d_off   d_name
 1052206  regular      32 5833893752366624558  getdents
 1048126  regular      40 8853463837756798l3  Operating System
 1052322  regular      56 1243422150728422115  Operating_System_2nd_Assignment2
.txt
 1051414  regular      40 1698968499403734134  Bahir Dar University
 1052108  regular      40 4227457652821878639  sytem_programming.c
 1052321  regular      56 5212049685710857374  Operating_System_2nd_Assignment2
 1047586  directory    24 5776565063095660060  .
 1049100  regular      40 6757211721830108338  System Programming
 1051446  regular      24 6911040291213905194  os.c
 1052323  regular      64 7296044588917893782  implemenation_using_different_fl
ag.txt
 1052324  regular      40 9176797301900980548  system_programming.c
 1046532  directory    24 9223372036854775807  ..
tade@tade-VirtualBox:~/Documents$ 
```

The source code using O_DIRECTORY flag

```c
#define _GNU_SOURCE
#include <dirent.h>
#include <fcntl.h>
#include <stdint.h>
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <sys/stat.h>
#include <sys/syscall.h>
#define handle_error(msg)
              do { perror(msg); exit(EXIT_FAILURE); } while (0)

       struct linux_dirent {
           unsigned long  d_ino;
           off_t          d_off;
           unsigned short d_reclen;
           char           d_name[];
```

```c
    };

    #define BUF_SIZE 1024

    int
    main(int argc, char *argv[])
    {
        int fd;
        long nread;
        char buf[BUF_SIZE];
        struct linux_dirent *d;
        char d_type;

        fd = open(argc > 1 ? argv[1] : ".", O_RDONLY);
        if (fd == -1)
            handle_error("open");

        for (;;) {
            nread = syscall(SYS_getdents, fd, buf, BUF_SIZE);
            if (nread == -1)
                handle_error("getdents");

            if (nread == 0)
                break;
            printf("--------------- nread=%d ---------------\n", nread);
            printf("inode#    file type  d_reclen  d_off   \t\t d_name\n");
            for (long bpos = 0; bpos < nread;) {
                d = (struct linux_dirent *) (buf + bpos);
                printf("%8ld  ",  d->d_ino);
                d_type = *(buf + bpos + d->d_reclen - 1);
                printf("%-10s ", (d_type == DT_REG) ?  "regular" :
                                 (d_type == DT_DIR) ?  "directory" :
                                 (d_type == DT_FIFO) ? "FIFO" :
                                 (d_type == DT_SOCK) ? "socket" :
                                 (d_type == DT_LNK) ?  "symlink" :
                                 (d_type == DT_BLK) ?  "block dev" :
                                 (d_type == DT_CHR) ?  "char dev" : "???");
                printf("%4d %10jd  %s\n", d->d_reclen,
                        (intmax_t) d->d_off, d->d_name);
                bpos += d->d_reclen;
            }
        }
        exit(EXIT_SUCCESS);
}
```

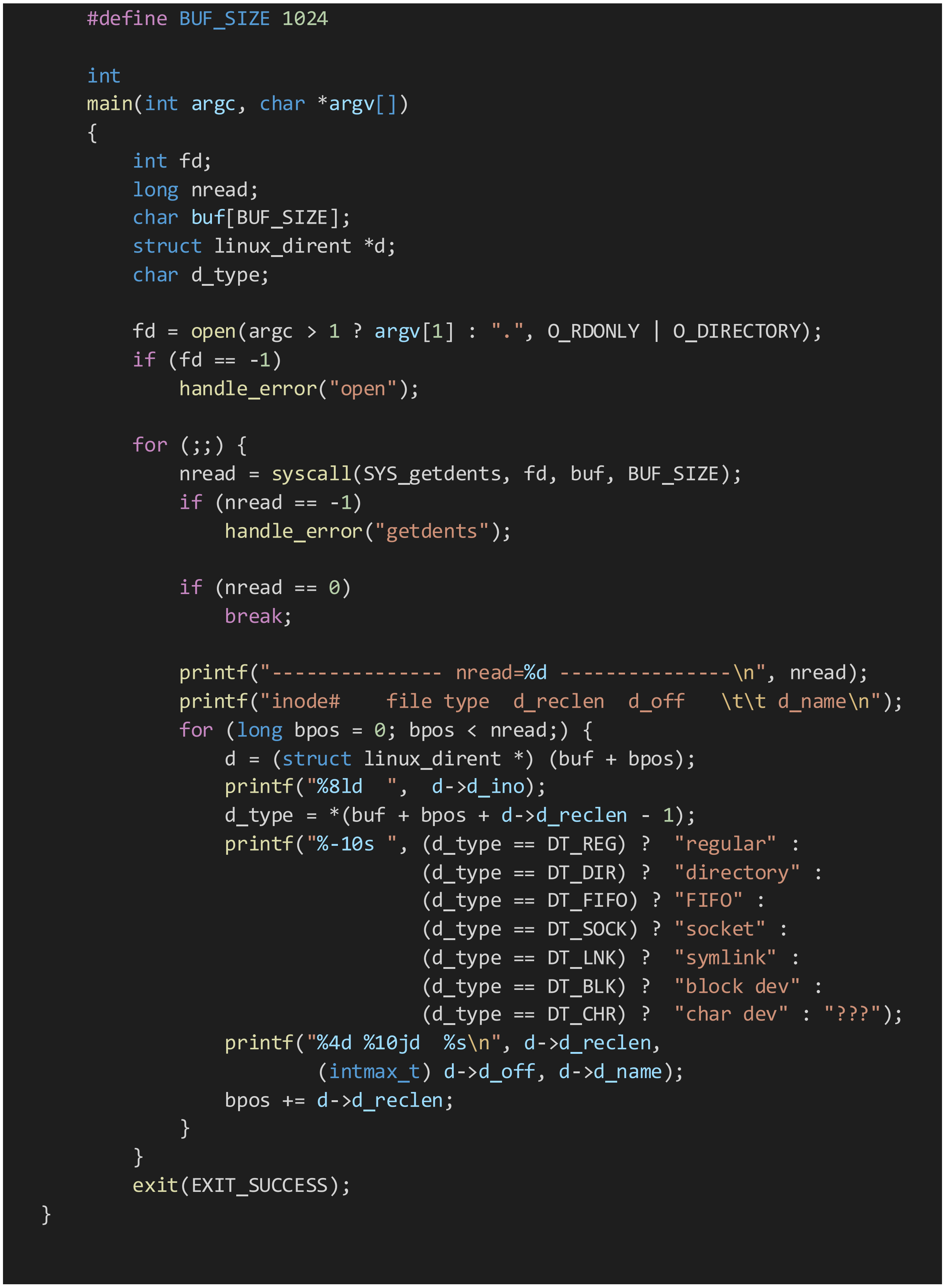
The output of the above source code using O_DIRECTORY



```
tade@tade-VirtualBox:~/Documents$ nano system_programming.c
tade@tade-VirtualBox:~/Documents$ gcc system_programming.c -o software
tade@tade-VirtualBox:~/Documents$ ls
'Bahir Dar University'                         os.c
 getdents                                      software
 implemenation_using_different_flag.txt      'System Programming'
'Operating System'                             system_programming.c
 Operating_System_2nd_Assignment2              sytem_programming.c
 Operating_System_2nd_Assignment2.txt          tadele
tade@tade-VirtualBox:~/Documents$ ./software
--------------- nread=544 ---------------
i-node#  file type  d_reclen  d_off    d_name
 1052204  regular      32 5833893752366245558  getdents
 1048126  regular      40 8853463837756798813  Operating System
 1052322  regular      56 1243422150728422115  Operating_System_2nd_Assignment2
.txt
 1051414  regular      40 1421852670866988034  Bahir Dar University
 1052212  regular      32 1698968499403734134  software
 1052108  regular      40 4227457652821878639  sytem_programming.c
 1052321  regular      56 4751418523351012335  Operating_System_2nd_Assignment2
 1052205  regular      32 5212049685710857374  tadele
 1047586  directory    24 5776565063095660060  .
 1049100  regular      40 6757211721830108338  System Programming
 1051446  regular      24 6911040291213905194  os.c
 1052323  regular      64 7296044588917893782  implemenation_using_different_fl
ag.txt
 1052324  regular      40 9176797301900980548  system_programming.c
 1046532  directory    24 9223372036854775807  ..
tade@tade-VirtualBox:~/Documents$
```

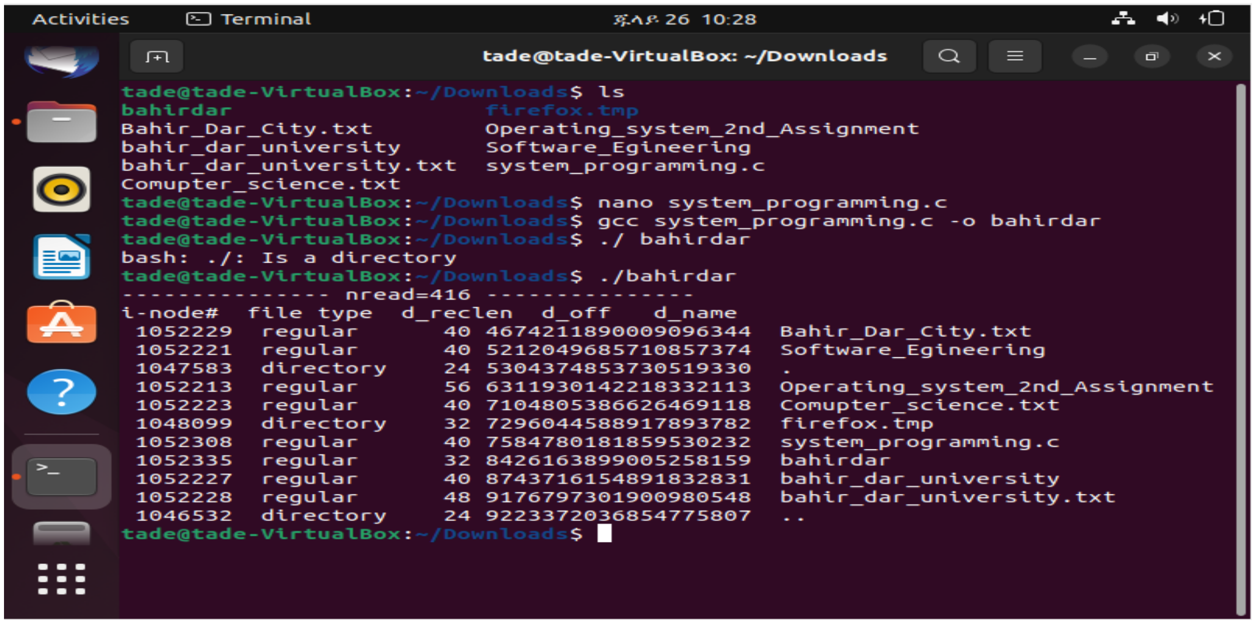
Source code implementation by using both flags at the same time.

```
#define _GNU_SOURCE
#include <dirent.h>
#include <fcntl.h>
#include <stdint.h>
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <sys/stat.h>
#include <sys/syscall.h>
#define handle_error(msg)
               do { perror(msg); exit(EXIT_FAILURE); } while (0)

       struct linux_dirent {
           unsigned long  d_ino;
           off_t          d_off;
           unsigned short d_reclen;
           char           d_name[];
       };
```



```c
    #define BUF_SIZE 1024

    int
    main(int argc, char *argv[])
    {
        int fd;
        long nread;
        char buf[BUF_SIZE];
        struct linux_dirent *d;
        char d_type;

        fd = open(argc > 1 ? argv[1] : ".", O_RDONLY | O_DIRECTORY);
        if (fd == -1)
            handle_error("open");

        for (;;) {
            nread = syscall(SYS_getdents, fd, buf, BUF_SIZE);
            if (nread == -1)
                handle_error("getdents");

            if (nread == 0)
                break;

            printf("--------------- nread=%d ---------------\n", nread);
            printf("inode#    file type  d_reclen  d_off   \t\t d_name\n");
            for (long bpos = 0; bpos < nread;) {
                d = (struct linux_dirent *) (buf + bpos);
                printf("%8ld  ",  d->d_ino);
                d_type = *(buf + bpos + d->d_reclen - 1);
                printf("%-10s ", (d_type == DT_REG) ?  "regular" :
                                 (d_type == DT_DIR) ?  "directory" :
                                 (d_type == DT_FIFO) ? "FIFO" :
                                 (d_type == DT_SOCK) ? "socket" :
                                 (d_type == DT_LNK) ?  "symlink" :
                                 (d_type == DT_BLK) ?  "block dev" :
                                 (d_type == DT_CHR) ?  "char dev" : "???");
                printf("%4d %10jd  %s\n", d->d_reclen,
                        (intmax_t) d->d_off, d->d_name);
                bpos += d->d_reclen;
            }
        }
        exit(EXIT_SUCCESS);
}
```

The output of the above source code:



```
tade@tade-VirtualBox:~/Downloads$ ls
bahirdar                  firefox.tmp
Bahir_Dar_City.txt        Operating_system_2nd_Assignment
bahir_dar_university      Software_Egineering
bahir_dar_university.txt  system_programming.c
Comupter_science.txt
tade@tade-VirtualBox:~/Downloads$ nano system_programming.c
tade@tade-VirtualBox:~/Downloads$ gcc system_programming.c -o bahirdar
tade@tade-VirtualBox:~/Downloads$ ./ bahirdar
bash: ./: Is a directory
tade@tade-VirtualBox:~/Downloads$ ./bahirdar
--------------- nread=416 ---------------
i-node#  file type  d_reclen  d_off    d_name
 1052229  regular      40 4674211890009096344  Bahir_Dar_City.txt
 1052221  regular      40 5212049685710857374  Software_Egineering
 1047583  directory    24 5304374853730519330  .
 1052213  regular      56 6311930142218332113  Operating_system_2nd_Assignment
 1052223  regular      40 7104805386626469118  Comupter_science.txt
 1048099  directory    32 7296044588917893782  firefox.tmp
 1052308  regular      40 7584780181859530232  system_programming.c
 1052335  regular      32 8426163899005258159  bahirdar
 1052227  regular      40 8743716154891832831  bahir_dar_university
 1052228  regular      48 9176797301900980548  bahir_dar_university.txt
 1046532  directory    24 9223372036854775807  ..
tade@tade-VirtualBox:~/Downloads$ 
```

REFERENCE:

The Linux and Unix System programming Handbook
https://man7.org/linux/man-pages/man2/getdents.2.html.
https://linuxhint.com/c-language-o_donly-o_wrongly-and-o_rdwr-flags/