Упражнение 12

Системни примитиви за работа с файлове в Linux

Как да компилираме С програма?

```
cc -o file.exe file.c
```

Какво е системен примитив?

Системните примитиви (системни извиквания (system calls)) са фундаменталния интерфейс между програма и ядрото на Linux-а (ядрото е програма, която предоставя основни услуги на другите програми). Т.е чрез системните примитиви ядрото предоставя услуги на приложните програми.

Защо на С?

Освен, че Linux е написан на С, за всеки системен примитив има поне една функция в стандартната библиотека на С. Ние ще пишем програми, които съдържат обръщения към тези функции.

Файлов дескриптор - Неотрицателно цяло число, което служи за уникален идентификатор на отворен файл. Файловите дескриптори имат локално значение за процесите – всеки процес разполага с N файлови дескриптора. Всеки процес стандартно има отворени три дескриптора:

- 0 (стандартен вход)
- 1 (стандартен изход)
- 2 (стандартен изход за грешки)

Отваряне и създаване на файл

```
#include <fcntl.h>
int open(const char *filename, int oflag [, mode_t mode]);
```

- Отваря файл с име **filename** в режим **oflag**, а ако не съществува го създава с режим **mode**.
- връща файлов дескриптор, с който се работи в програмата или -1 при неуспех

oflag указва режима на отваряне и/или действия, които да се извършат след отваряне:

- O_RDONLY - само за четене
- O_WRONLY - само за писане
- O_RDWR - за четене и писане
- O_CREAT - създава файла

- O_EXCL - заедно с O_CREAT: ако файлът **не съществува**, той се създава, иначе се връща грешка
- O_TRUNC - старото съдържание на файла се изтрива след отваряне
- O_APPEND - старото съдържание се запазва, а новото се вписва в края на файла

Четене от файл

```
#include <unistd.h>
ssize_t read(int fd, void *buf, size_t nbytes);
```

- **fd** - номер на файлов дескриптор
- **buf** - указател към областта от програмата, където се записват данните
- **nbytes** - указва броя байтове за четене
- връща броя на **реално** прочетените байтове, иначе връща **-1**
- при край на файл - **0**

Писане във файл

```
ssize_t write(int fd, void *buf, size_t nbytes);
```

- **fd** - номер на файлов дескриптор
- **buf** - указател към областта от програмата, където се записват данните
- **nbytes** - указва броя байтове за писане
- връща броя на **реално** записаните байтове, иначе връща -1

Позициониране във файл

```
off_t lseek(int fd, off_t offset, int whence);
```

- **fd** - номер на файлов дескриптор
- **offset** - самото отместване
- **whence** - указва как се интерпретира отместването
  - **SEEK_SET** - премества указателя от началото до offset-байта
  - **SEEK_CUR** - премества указателя от сегашната му позиция до offset-байта
  - **SEEK_END** - премества указателя от края на файла (отзад-напред) до offset-байта (отрицателно число)

Затваряне на файл

```
int close(int fd);
```