open(2)

формат: int open(const char *pathname, int flags);
библиотека: fcntl.h
използва се за: отваряне и може би създаване на файл
допълнителна информация:
Видове flags:
O_CREAT - ако файла не съществува създай го;
O_RDONLY - read-only достъп;
O_WRONLY - write-only достъп;
O_RDWR - read/write достъп.
O_TRUNC - the file to be truncated if it exists.
O_APPEND - writes to append to the end of the file instead of overwrite at the start.

close(2)

формат: int close(int fd);
библиотека: unistd.h
използва се за: затваряне на файл
допълнителна информация:
Връща 0 за успешно затваряне и -1 при грешка

malloc(3)

формат: void *malloc(size_t size);
библиотека: stdlib.h
използва се за: отделяне на дадена памет и връща пойнтер към нея;
допълнителна информация:
Използвай free след приключване на действието с паметта;

free(3)

формат: void free(void *ptr);
библиотека: stdlib.h
използва се за: освобождаване на паметта в пойтер
допълнителна информация:

exit(3)

формат: void exit(int status);
библиотека: stdlib.h
използва се за: край на програмата
допълнителна информация:

read(2)

формат: ssize_t read(int fd, void *buf, size_t count)
библиотека: unistd.h
използва се за: чете от файл
допълнителна информация:
Чете count байта от файл fd в буфер buf.

write(2)

формат: ssize_t write(int fd, void *buf, size_t count)
библиотека: unistd.h
използва се за: писане във файл
допълнителна информация:
Пише count байта във файл fd от буфер buf.

uintX_t

библиотека: stdint.h

използва се за: Видове unsigned integers

допълнителна информация:

X- дължина в битове

err(3)

формат: void err(int eval, const char *fmt,...);

библиотека: err.h

използва се за: извеждане на съобщение за грешка в stderr

допълнителна информация:

atoi(3)

формат: int atoi(const char *nptr);

библиотека: stdlib.h

използва се за: преобразува string to a integer

допълнителна информация:

lseek(2)

формат: off_t lseek(int fd, off_t offset, int whence);

библиотека: unistd.h

използва се за: сменяне на местоположението на read/write offset

допълнителна информация:

Видове whence:

SEEK_SET - offset на файла е поставен на offset bytes.

SEEK_CUR - offset на файла е поставен сегашноте местополож. + стойноста на offset bytes.

SEEK_END - offset на файла е поставен на размера на файла + стойноста offset bytes.

Функция за бит на даден индекс:

```
bool bit( uintX_t num, int i){
    return ((( num >> i) & 1) == 1);
}
```

xxd -b <filename> извежда бинарен файл с бинарни стойности;

execl(3)

формат: int execl(const char *path, const char *arg, ... /* (char *) NULL */);

библиотека: unistd.h

използва се за:

The exec() family of functions replaces the current process image with a new process image.

допълнителна информация:

The exec() functions only return if an error has occurred.

The return value is -1, and errno is set to indicate the error.

execlp(3)

формат: int execlp(const char *file, const char *arg, ... /* (char *) NULL */);

библиотека: unistd.h

използва се за:

The execlp(), execvp(), and execvpe() functions duplicate the actions of the shell in searching for an executable file if the specified filename does not contain a slash (/) character.

допълнителна информация:

fork(2)

формат: pid_t fork(void);
библиотека: unistd.h
използва се за: create a child process
допълнителна информация:
the child inherits copies of the parent's set of open file descriptors

wait(2)

формат: pid_t wait(int *wstatus);
библиотека: sys/wait.h
използва се за: wait for process to change state
допълнителна информация:
All of these system calls are used to wait for state changes in a child of the calling process, and obtain information about the child whose state has changed.
A state change is considered to be:
the child terminated;
the child was stopped by a signal;
or the child was resumed by a signal.

getpid(2)

формат: pid_t getpid(void); pid_t getppid(void);
библиотека: unistd.h
използва се за: get process identification
допълнителна информация:
- getpid() returns the process ID (PID) of the calling process.(This is often used by routines that generate unique temporary filenames.)
- getppid() returns the process ID of the parent of the calling process.

pipe(7)

формат:
библиотека:
използва се за: overview of pipes and FIFOs
допълнителна информация:

pipe(2)

формат: int pipe(int pipefd[2]);
библиотека: unistd.h
използва се за: create pipe
допълнителна информация:
On success, zero is returned.
On error, -1 is returned.

mkfifo(3)

формат: int mkfifo(const char * pathname, mode_t mode);
библиотека: <sys/types.h> ; <sys/stat.h>
използва се за: make a FIFO special file (a named pipe)
допълнителна информация:
- makes a FIFO special file with name pathname. mode specifies the FIFO's permissions.

On success mkfifo() and mkfifoat() return 0.
On error, -1 is returned

dup(2)

формат: int dup(int oldfd);

библиотека: unistd.h

използва се за: duplicate a file descriptor

допълнителна информация:

The dup() system call allocates a new file descriptor that refers to the same open file description as the descriptor oldfd. The new file descriptor number is guaranteed to be the lowest-numbered file descriptor that was unused in the calling process.

dup2(2)

формат: int dup2(int oldfd, int newfd);

библиотека: unistd.h

използва се за: duplicate a file descriptor

допълнителна информация:

The dup2() system call performs the same task as dup(), but instead of using the lowest-numbered unused file descriptor, it uses the file descriptor number specified in newfd. In other words, the file descriptor newfd is adjusted so that it now refers to the same open file description as oldfd.

Библиотеки:

<fcntl.h>
<unistd.h>
<stdlib.h>
<stdint.h>
<stdbool.h>
<err.h>
<wait.h>
<stat.h>