

atoi(): int atoi(const char *str);

- Converts the string arg str to an integer

chdir(): int chdir(const char *path);

- Changes the current working directory to path, which can be relative to the current working directory or an absolute path name.

fopen(): FILE *fopen(const char *filename, const char *mode);

- Opens the filename pointed to, by filename using the given mode.
- Modes: r, w, a, r+, w+, a+

fclose(): int fclose(FILE *stream);

- Closes the stream. All buffers are flushed.

perror(): void perror(const char *str);

- Prints a descriptive error message to stderr. First, the string str is printed, followed by a colon then a space.
- Ex: perror("Error: ");

signal(): void (*signal(int sig, void (*func)(int)))(int);

- Sets a function to handle signal i.e. a signal handler with signal number sig.
- Ex:

```
void sighandler(int);
int main () {
    signal(SIGINT, sighandler);

    while(1) {
        printf("Going to sleep for second...\n");
        sleep(1); //slep(*int) int is in second
    }
    return(0);
}

void sighandler(int signum) {
    printf("Caught signal %d, coming out...\n", signum);
    exit(1);
}
```

exit(): void exit(int status);

- Terminates the calling process immediately. Status value return to the parent process.

htons(): The htons() function makes sure that numbers are stored in memory in network byte order, which is with the most significant byte first.

- Ex: if the number is 0x1389 in hexadecimal, so the bytes involved are 0x13 and 0x8

- Little-endian (from the computer): htons() will convert to 0x89 and 0x13
- Big-endian (from the computer): htons() will not do anything, still be the same as original 0x13 0x89

socket(): int socket(int domain, int type, int protocol);

- Create an unbound socket in a communications domain, and return a file descriptor that can be used in later function calls that operate on sockets.
- <https://pubs.opengroup.org/onlinepubs/009604499/functions/socket.html>

bind(): int bind(int socket, const struct sockaddr *address, socklen_t address_len);

- Assign a local socket address address to a socket identified by descriptor socket that has no local socket address assigned. Sockets created with the socket() function are initially unnamed; they are identified only by their address family.
- <https://pubs.opengroup.org/onlinepubs/009695399/functions/bind.html>

listen(): int listen(int sockfd, int backlog);

- Marks the socket referred to by sockfd as a passive socket, that is, as a socket that will be used to accept incoming connection requests using accept(2).
- <https://man7.org/linux/man-pages/man2/listen.2.html>

accept(): int accept(int sockfd, struct sockaddr *restrict addr, socklen_t *restrict addrlen);

- The accept() system call is used with connection-based socket types (SOCK_STREAM, SOCK_SEQPACKET). It extracts the first connection request on the queue of pending connections for the listening socket, sockfd, creates a new connected socket, and returns a new file descriptor referring to that socket. The newly created socket is not in the listening state. The original socket sockfd is unaffected by this call.
- <https://man7.org/linux/man-pages/man2/accept.2.html>