**LAB ASSIGNMENT - 4**

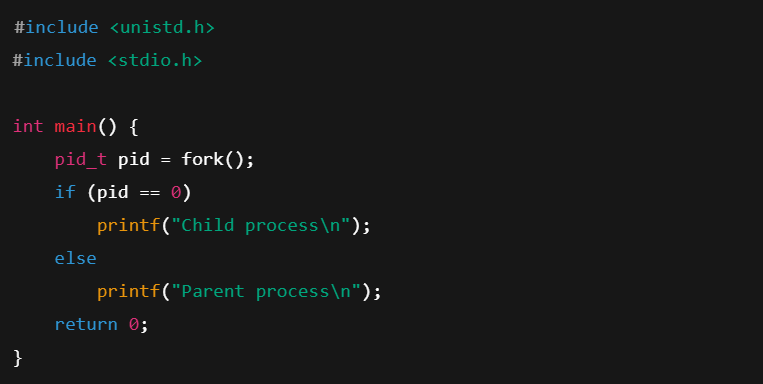
**System Calls**

**1. Process Management System Calls**

**(i) fork() -**

* A new process is created by the fork() system call.
* A new process may be created with fork() without a new program being run-the new sub-process simply continues to execute exactly the same program that the first (parent) process was running.
* It is one of the most widely used system calls under process management.
* Creates a child process that runs concurrently with the parent process. It returns:
* 0 to the child process
* Child’s PID to the parent process
* -1 if creation fails

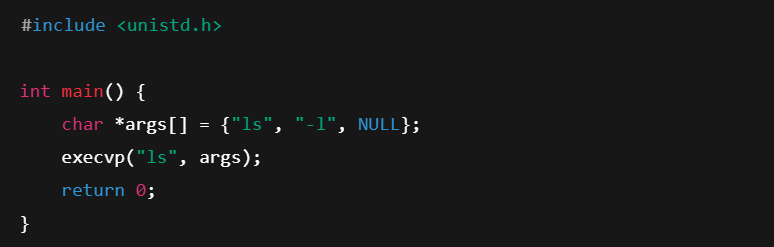
**Example:**



**(ii) exec() -**

* A new program will start executing after a call to exec()
* Running a new program does not require that a new process be created first: any process may call exec() at any time. The currently running program is immediately terminated, and the new program starts executing in the context of the existing process.
* Replaces the current process image with a new process image.

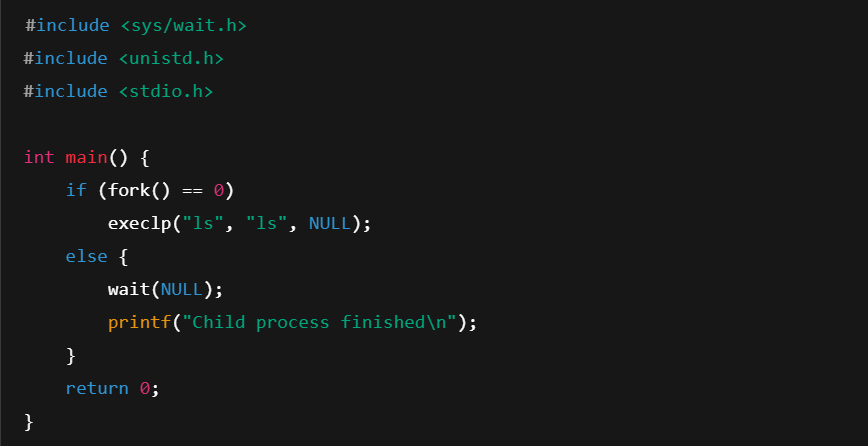
**Example;**



**(iii) wait() -**

* Makes the parent process wait until a child process terminates.

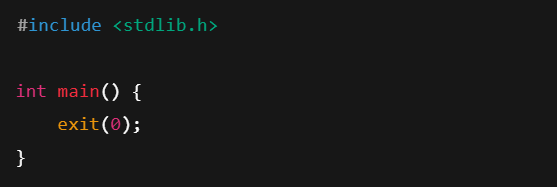
**Example:**



**(iv) exit -**

* The exit() system call is used by a program to terminate its execution.
* The operating system reclaims resources that were used by the process after the exit() system call.

**Example:**

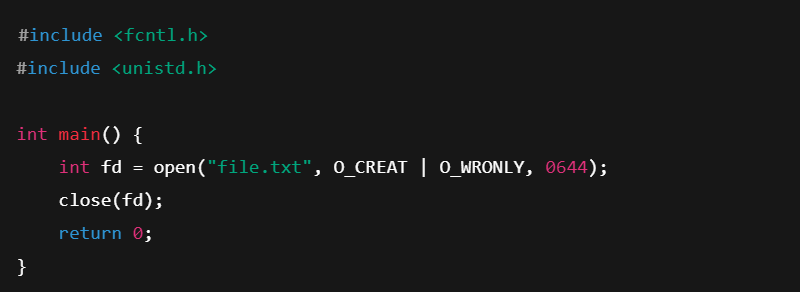


**2. File Management System Calls**

**(i) open() -**

* It is the system call to open a file.
* This system call just opens the file, to perform operations such as read and write, we need to execute different system call to perform the operations.

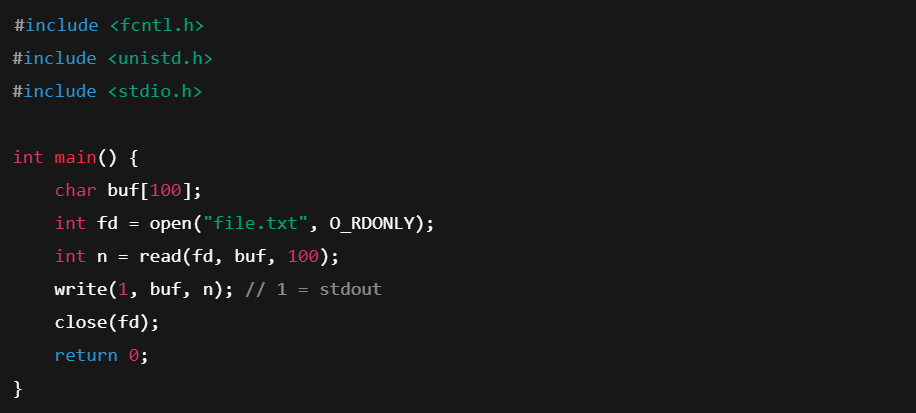
**Example:**

****

**(ii) read() -**

* + This system call opens the file in reading mode
  + We can not edit the files with this system call.
  + Multiple processes can execute the read() system call on the same file simultaneously.

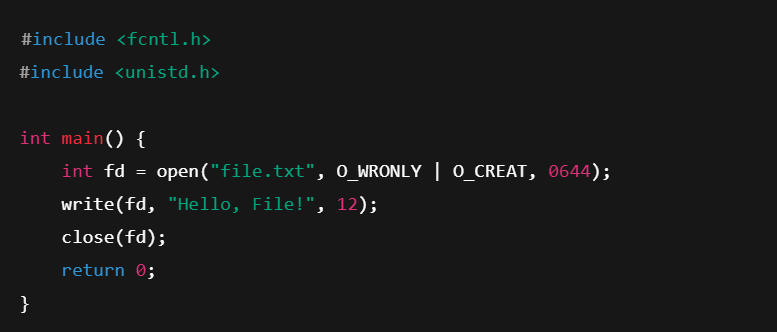
**Example:**

****

**(iii) write() -**

* + This system call opens the file in writing mode
  + We can edit the files with this system call.
  + Multiple processes can not execute the write() system call on the same file simultaneously.

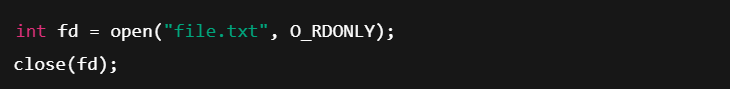
**Example:**

****

**(iv) close() -**

* This system call closes the opened file.

**Example:**

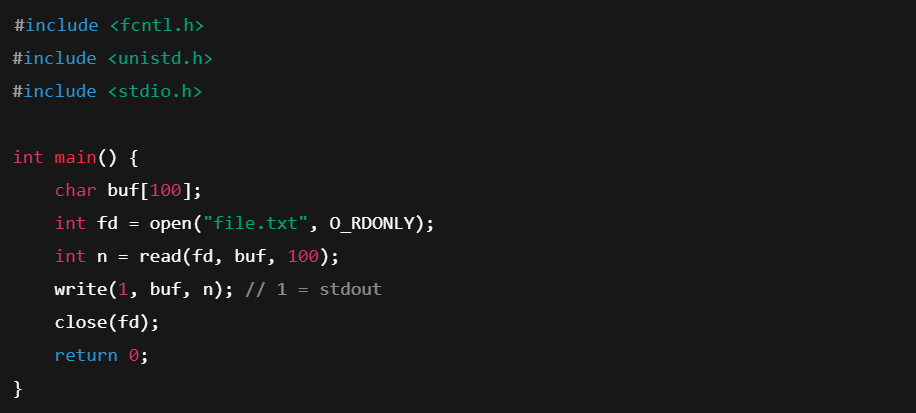
****

**3. Device Management System calls**

**(i) read() -**

* + This system call opens the file in reading mode
  + We can not edit the files with this system call.
  + Multiple processes can execute the read() system call on the same file simultaneously.

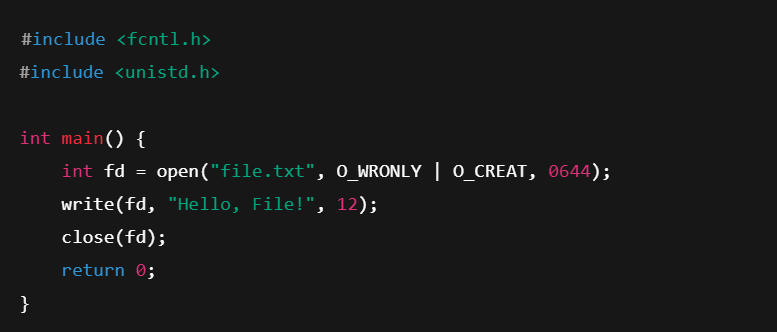
**Example:**

****

**(ii) write() -**

* + This system call opens the file in writing mode
  + We can edit the files with this system call.
  + Multiple processes can not execute the write() system call on the same file simultaneously.

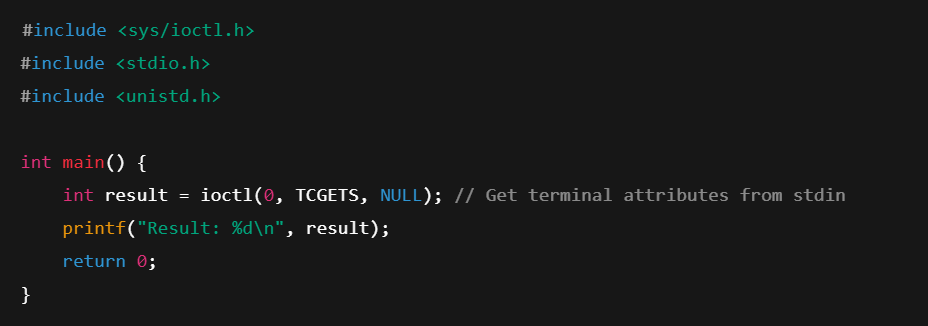
**Example:**

****

**(iii) ioctl() -**

* ioctl() is referred to as Input and Output Control.
* ioctl is a system call for device-specific input/output operations and other operations which cannot be expressed by regular system calls.

**Example:**

****

**(iv) select() -**

* Monitors multiple file descriptors to see if I/O is possible.

**Example:**

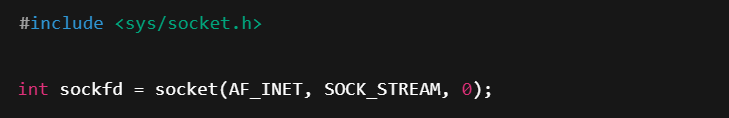
****

**4. Network Management System Calls**

**(i) socket() -**

* Creates a new socket.

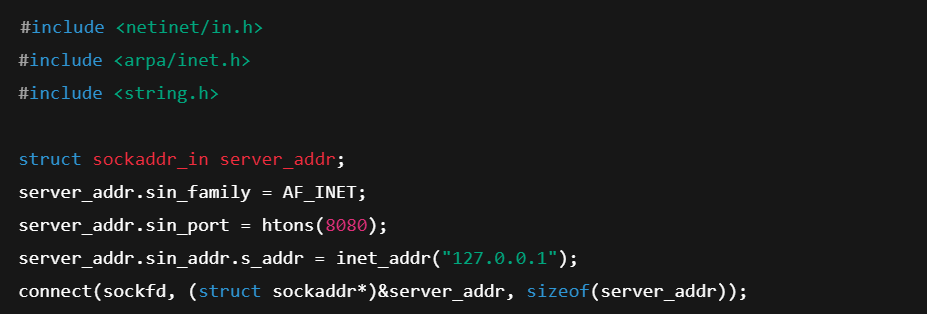
**Example:**

****

**(ii) connect() -**

* Establishes a connection to a server.

**Example:**

****

**(iii) send() -**

* Used to send messages on a socket.

**Example:**

****

**(iv) recv() -**

* Used to receive messages on a socket.

**Example:**

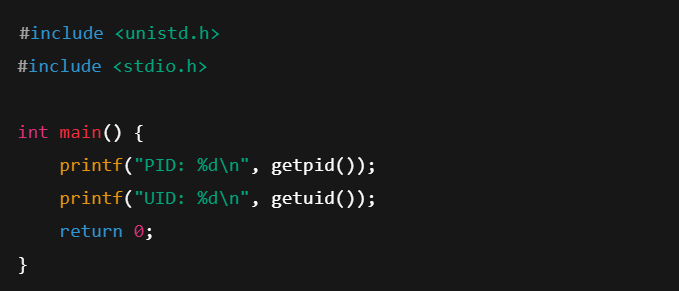
****

**5. System Information Management System Calls**

**(i) getpid() -**

* getpid stands for Get the Process ID.
* The getpid() function shall return the process ID of the calling process.
* The getpid() function shall always be successful and no return value is reserved to indicate an error.

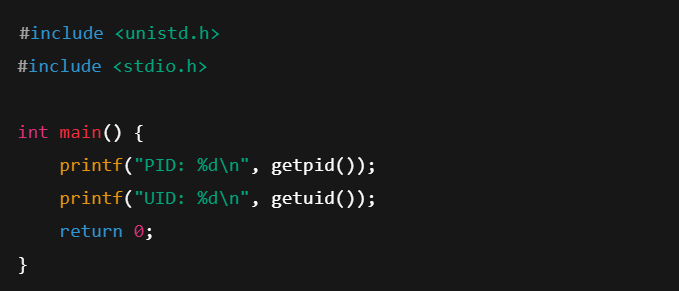
**Example:**

****

**(ii) getuid() -**

* getuid stands for Get the User ID.
* The getuid() function shall return the User ID of the calling process.
* The getuid() function shall always be successful and no return value is reserved to indicate an error.

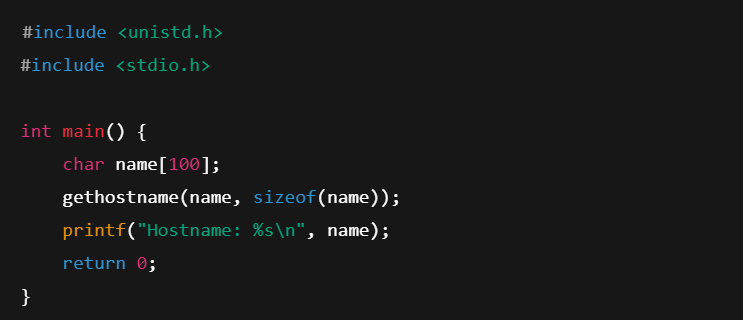
**Example:**

****

**(iii) gethostname() -**

* gethostname stands for Get the hostname of the system.
* The gethostname() function shall always be successful and no return value is reserved to indicate an error.

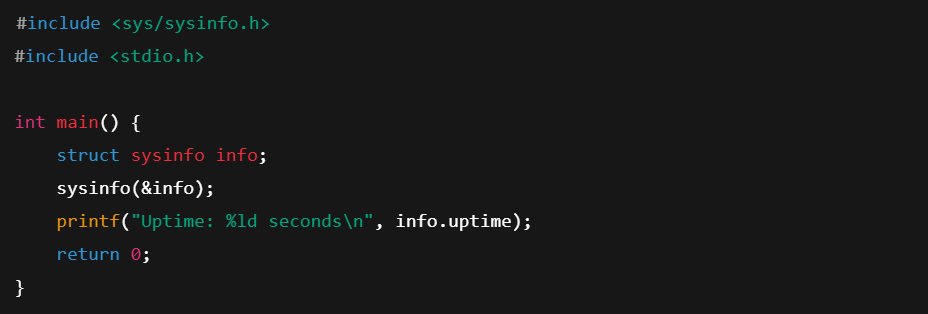
**Example:**

****

**(iv) sysinfo() -**

* Returns system-wide information.

**Example:**

****