**Lab Assignment 4**

**To perform: Comprehensive study of different categories of Linux system calls, categorized as**

**Q1. Process Management System Calls**

| **System Call** | **Description** |
| --- | --- |
| **fork()** | **Creates a new process by duplicating the current process. The new process is called a child process.** |
| **exec()** | **Replaces the current process image with a new process image (used to run a different program).** |
| **wait()** | **Makes the parent process wait until its child process finishes execution.** |
| **exit()** | **Terminates the current process and returns a status code to the parent process.** |

**💡 Example:**

**c**

**pid\_t pid = fork();**

**if (pid == 0) {**

**execl("/bin/ls", "ls", NULL);**

**exit(0);**

**} else {**

**wait(NULL);**

**}**

**Q2. File Management System Calls**

| **System Call** | **Description** |
| --- | --- |
| **open()** | **Opens a file and returns a file descriptor.** |
| **read()** | **Reads data from a file into a buffer.** |
| **write()** | **Writes data from a buffer to a file.** |
| **close()** | **Closes an opened file descriptor.** |

**💡 Example:**

**c**

**int fd = open("file.txt", O\_RDONLY);**

**read(fd, buffer, 100);**

**close(fd);**

**Q3. Device Management System Calls**

| **System Call** | **Description** |
| --- | --- |
| **read()** | **Reads data from a device (like a keyboard or file).** |
| **write()** | **Writes data to a device.** |
| **ioctl()** | **Performs device-specific input/output operations.** |
| **select()** | **Monitors multiple file descriptors to see if I/O is possible on any of them.** |

**💡 Example:**

**c**

**ioctl(fd, COMMAND, &arg);**

**Q4. Network Management System Calls**

| **System Call** | **Description** |
| --- | --- |
| **socket()** | **Creates a new communication endpoint (socket).** |
| **connect()** | **Connects a socket to a remote server address.** |
| **send()** | **Sends data through a socket.** |
| **recv()** | **Receives data from a socket.** |

**💡 Example:**

**c**

**int sock = socket(AF\_INET, SOCK\_STREAM, 0);**

**connect(sock, (struct sockaddr\*)&server, sizeof(server));**

**send(sock, "Hello", strlen("Hello"), 0);**

**recv(sock, buffer, sizeof(buffer), 0);**

**Q5. System Information Management System Calls**

| **System Call** | **Description** |
| --- | --- |
| **getpid()** | **Returns the Process ID (PID) of the current process.** |
| **getuid()** | **Returns the User ID (UID) of the process owner.** |
| **gethostname()** | **Gets the name of the host (machine).** |
| **sysinfo()** | **Returns system-wide statistics like uptime, memory usage, etc.** |

**💡 Example:**

**c**

**pid\_t pid = getpid();**

**uid\_t uid = getuid();**

**char hostname[100];**

**gethostname(hostname, sizeof(hostname));**