# Aim

To study and learn about various system calls in Linux.

# To Perform

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

# Process Management System Calls

* + fork(): Used to create a new process by duplicating the calling process.
  + exec(): Replaces the current process image with a new process image.
  + wait(): Makes a process wait until its child process finishes execution.
  + exit(): Terminates the calling process.

Example:

#include <stdio.h> #include <unistd.h> #include <sys/wait.h> int main() {

pid\_t pid = fork(); if (pid == 0) {

printf("Child Process\n"); execlp("/bin/ls", "ls", NULL);

} else {

wait(NULL);

printf("Parent Process\n");

}

return 0;

}

# File Management System Calls

* + open(): Opens a file.
  + read(): Reads data from a file.
  + write(): Writes data to a file.
  + close(): Closes an open file.

Example:

#include <fcntl.h> #include <unistd.h> int main() {

int fd = open("test.txt", O\_WRONLY | O\_CREAT, 0644); write(fd, "Hello, World!", 13);

close(fd); return 0;

}

# Device Management System Calls

* + read(), write(): Same as file operations, used for reading/writing to devices.
  + ioctl(): Device-specific input/output operations.
  + select(): Monitors multiple file descriptors.

Example:

#include <stdio.h> #include <sys/ioctl.h> #include <fcntl.h> #include <unistd.h> int main() {

int fd = open("/dev/tty", O\_RDONLY); if (fd != -1) {

int bytes;

ioctl(fd, FIONREAD, &bytes); printf("Bytes available: %d\n", bytes); close(fd);

}

return 0;

}

# Network Management System Calls

* + socket(): Creates a socket.
  + connect(): Connects the socket to a remote address.
  + send(): Sends data through a socket.
  + recv(): Receives data from a socket.

Example:

#include <stdio.h> #include <string.h> #include <sys/socket.h> #include <arpa/inet.h> int main() {

int sock = socket(AF\_INET, SOCK\_STREAM, 0); struct sockaddr\_in server; server.sin\_addr.s\_addr = inet\_addr("127.0.0.1"); server.sin\_family = AF\_INET;

server.sin\_port = htons(8080);

connect(sock, (struct sockaddr \*)&server, sizeof(server)); send(sock, "Hello", strlen("Hello"), 0);

char buffer[1024]; recv(sock, buffer, 1024, 0);

printf("Received: %s\n", buffer); return 0;

}

# System Information Management System Calls

* + getpid(): Gets the process ID.
  + getuid(): Gets the user ID.
  + gethostname(): Gets the host name of the machine.
  + sysinfo(): Retrieves overall system statistics.

Example:

#include <stdio.h> #include <unistd.h> #include <sys/sysinfo.h> int main() {

printf("PID: %d\n", getpid());

printf("UID: %d\n", getuid());

char hostname[1024]; gethostname(hostname, sizeof(hostname)); printf("Hostname: %s\n", hostname);

struct sysinfo info; sysinfo(&info);

printf("Uptime: %ld\n", info.uptime); return 0;

}