Experiment Number: 4

NAME: Omkar Sunil Khanvilkar ROLLNO: 07

CLASS: IT-B BATCH: 2

DATE OF PERFORMANCE: 20-8-2024

Problem Statement:

Design menu driven application demonstrating use of different system calls.

1. process related system call:fork,exit, wait,

2) file realted system call: open,read,write,close,link,unlink,stat

3) communication system call:pipe,fifo,

4)information related system call

Code:

```c
#include <stdio.h>

#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <fcntl.h>
#include <sys/stat.h>
#include <sys/utsname.h>

void process_related() {
    pid_t pid = fork();
    if (pid == 0) {
        printf("Child Process: PID = %d\n", getpid());
        exit(0);
    } else if (pid > 0) {
        wait(NULL);
        printf("Parent Process: PID = %d, Child Process has terminated\n",
getpid());
    } else {
        perror("fork failed");
    }
}

void file_related() {
    int fd = open("example.txt", O_CREAT | O_WRONLY, 0644);
    if (fd == -1) {
        perror("open failed");
        return;
```

```
    }
    write(fd, "Hello, World!\n", 14);
    close(fd);

    struct stat fileStat;
    if (stat("example.txt", &fileStat) < 0) {
        perror("stat failed");
        return;
    }
    printf("File Size: %ld bytes\n", fileStat.st_size);
    unlink("example.txt");
}

void communication_related() {
    int fd[2];
    char buffer[20];

    if (pipe(fd) == -1) {
        perror("pipe failed");
        return;
    }

    pid_t pid = fork();
    if (pid == 0) {
        close(fd[0]);
        write(fd[1], "Message from Child", 18);
        close(fd[1]);
        exit(0);
    } else {
        wait(NULL);
        close(fd[1]);
        read(fd[0], buffer, 18);
        printf("Parent received: %s\n", buffer);
        close(fd[0]);
    }
}

void info_related() {
    printf("Process ID: %d\n", getpid());
    printf("Parent Process ID: %d\n", getppid());
    printf("User ID: %d\n", getuid());
    printf("Group ID: %d\n", getgid());

    struct utsname unameData;
    if (uname(&unameData) < 0) {
        perror("uname failed");
        return;
    }
```

```c
    printf("System Information: %s\n", unameData.sysname);
}

int main() {
    int choice;
    while (1) {
        printf("\nMenu:\n");
        printf("1. Process Related System Calls\n");
        printf("2. File Related System Calls\n");
        printf("3. Communication Related System Calls\n");
        printf("4. Information Related System Calls\n");
        printf("5. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                process_related();
                break;
            case 2:
                file_related();
                break;
            case 3:
                communication_related();
                break;
            case 4:
                info_related();
                break;
            case 5:
                exit(0);
            default:
                printf("Invalid choice! Please try again.\n");
        }
    }
    return 0;
}
```

Output:
omkar@omkar-VirtualBox:~$ gcc -o system_calls system_calls.c
omkar@omkar-VirtualBox:~$ ./system_calls

Menu:
1. Process Related System Calls
2. File Related System Calls
3. Communication Related System Calls
4. Information Related System Calls
5. Exit

```
Enter your choice: 1
Child Process: PID = 4567
Parent Process: PID = 4566, Child Process has terminated

Menu:
1. Process Related System Calls
2. File Related System Calls
3. Communication Related System Calls
4. Information Related System Calls
5. Exit
Enter your choice: 2
File Size: 16 bytes

Menu:
1. Process Related System Calls
2. File Related System Calls
3. Communication Related System Calls
4. Information Related System Calls
5. Exit
Enter your choice: 3
Parent received: Message from Child


Menu:
1. Process Related System Calls
2. File Related System Calls
3. Communication Related System Calls
4. Information Related System Calls
5. Exit
Enter your choice: 4
Process ID: 4691
Parent Process ID: 4642
User ID: 1000
Group ID: 1000
System Information: Linux

Menu:
1. Process Related System Calls
2. File Related System Calls
3. Communication Related System Calls
4. Information Related System Calls
5. Exit
Enter your choice: 5
omkar@omkar-VirtualBox:~$
```