LAB – 4

AIM – Introduction to Processes

Shruti Mishra

21BCP110

1. C program to demonstrate system call are made 2^n time in fork

```c
#include<stdio.h>
#include<sys/types.h>
#include<unistd.h>

int main()
{
    fork();
    printf("Hello World\n");
    return 0;
}
```

Output :

```
shruti@shruti-VirtualBox:~/Documents/Fork_System_Call$ gcc -o ls program1.c
shruti@shruti-VirtualBox:~/Documents/Fork_System_Call$ ./ls
Hello World
Hello World
```

2. C program to demonstrate two process run at same time

```c
#include<stdio.h>
#include <stdlib.h>
#include<unistd.h>
#include<sys/types.h>

int main()
{
    fork();
```

```
    fork();
    fork();
    printf("hello\n");
    return 0;
}
```

Output :

```
shruti@shruti-VirtualBox:~/Documents/Fork_System_Call$ gcc -o ls program2.c
shruti@shruti-VirtualBox:~/Documents/Fork_System_Call$ ./ls
hello
hello
hello
hello
hello
hello
hello
hello
```

3. C program to demonstrate Parent process and child process

```
#include<stdio.h>
#include <unistd.h>

void forkexample()
{
    //child process because return value zero
    if (fork() == 0)
    {
        printf("Hello from child!\n");
    }
    //parent process because return value non-zero
    else{
        printf("Hello from parent!\n");
    }
}
int main()
{
```

```
    forkexample();
    return 0;
}
```

Output :

```
shruti@shruti-VirtualBox:~/Documents/Fork_System_Call$ gcc -o ls program3.c
shruti@shruti-VirtualBox:~/Documents/Fork_System_Call$ ./ls
Hello from child!
Hello from parent!
```

4. C program to demonstrate Orphan process

```
#include<stdio.h>
#include <stdlib.h>
#include<sys/types.h>
#include<unistd.h>

int main()
{
    int pid;
    pid = fork();
    if(pid==0)
    {
        printf("I Am The Child My Process ID Is %d\n",getpid());
        printf("My Parents Process ID Is %d\n",getppid());
        printf("Child Terminates\n");
        exit(0);
    }
    else{
        printf("I Am The Parent, My Process ID Is %d\n",getpid());
        printf("My Parent's Parent  Process ID Is %d\n",getppid());
        printf("Parent Terminates\n");
    }

    return 0;
}
```

Output :

```
shruti@shruti-VirtualBox:~/Documents/Fork_System_Call$ gcc -o ls orphan.c
shruti@shruti-VirtualBox:~/Documents/Fork_System_Call$ ./ls
I Am The Parent, My Process ID Is 2712
My Parent's Parent  Process ID Is 2368
Parent Terminates
I Am The Child My Process ID Is 2713
My Parents Process ID Is 2712
Child Terminates
```

5. C program to demonstrate zombie process

```c
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
#include <stdlib.h>
int main()
{
    //Fork Returns PID
    int child_pid = fork();

    //Parent Process
    if(child_pid>0)
    {
        sleep(60);
    }
    //Child Process
    else
    {
        exit(0);
    }
    return 0;
}
```

Output :

Program terminated after 60 sec (i.e. , parent sleeps for 60 sec).

```
shruti@shruti-VirtualBox:~/Documents/Fork_System_Call$ gcc -o ls zombie.c
shruti@shruti-VirtualBox:~/Documents/Fork_System_Call$ ./ls
shruti@shruti-VirtualBox:~/Documents/Fork_System_Call$
```

6. C program to demonstrate reparenting process

```c
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
#include <stdlib.h>

int main()
{
    //Fork Returns PID
    int child_pid = fork();

    int pid;
    pid = fork();

    if(pid==0)
    {
        printf("I Am The Child My Process ID Is %d\n",getpid());
        printf("My Parents Process ID Is %d\n",getppid());
        sleep(30);
        printf("After Sleep\n");
        printf("I Am The Parent, My Process ID Is %d\n",getpid());
        printf("I Parent's Parent, My Process ID Is %d\n",getppid());
        printf("Child Terminates\n");
    }

    else
```

```c
    {
        sleep(20);
        printf("I Am The Parent, My Process ID Is %d\n",getpid());
        printf("My Parent's Parent Process ID Is %d\n",getppid());
        printf("Parent Terminates\n");
    }
    return 0;
```

Output :

```
shruti@shruti-VirtualBox:~/Documents/Fork_System_Call$ gcc -o ls reparenting.c
shruti@shruti-VirtualBox:~/Documents/Fork_System_Call$ ./ls
I Am The Child My Process ID Is 2859
My Parents Process ID Is 2857
I Am The Child My Process ID Is 2860
My Parents Process ID Is 2858
I Am The Parent, My Process ID Is 2857
I Am The Parent, My Process ID Is 2858
My Parent's Parent Process ID Is 2857
My Parent's Parent Process ID Is 2368
Parent Terminates
Parent Terminates
shruti@shruti-VirtualBox:~/Documents/Fork_System_Call$ After Sleep
I Am The Parent, My Process ID Is 2859
I Parent's Parent, My Process ID Is 1592
Child Terminates
After Sleep
I Am The Parent, My Process ID Is 2860
I Parent's Parent, My Process ID Is 1592
Child Terminates
shruti@shruti-VirtualBox:~/Documents/Fork_System_Call$
```