OS-Lab

Lab#6

Name: Anas-Altaf

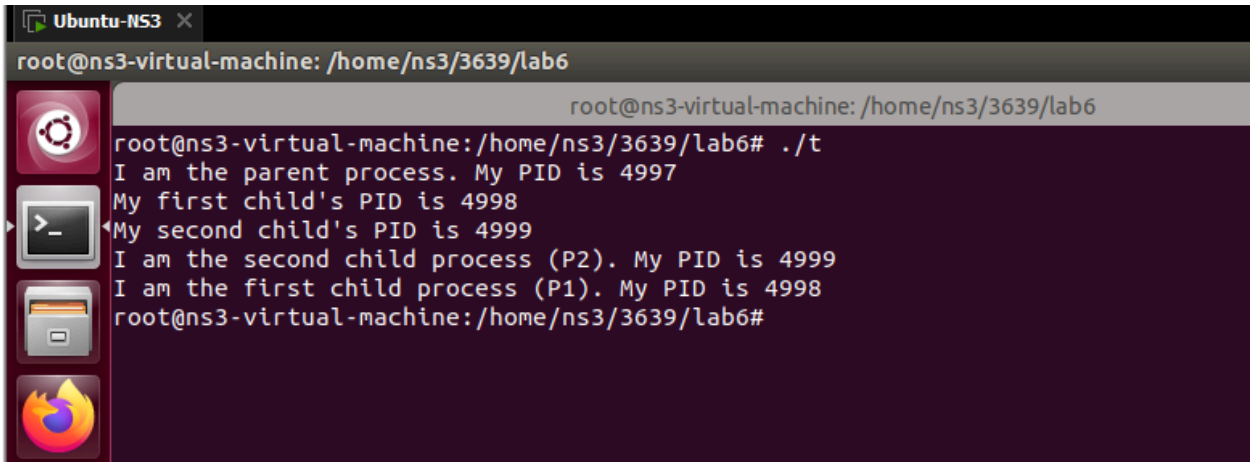
Roll.No: 22F-3639

C Codes:

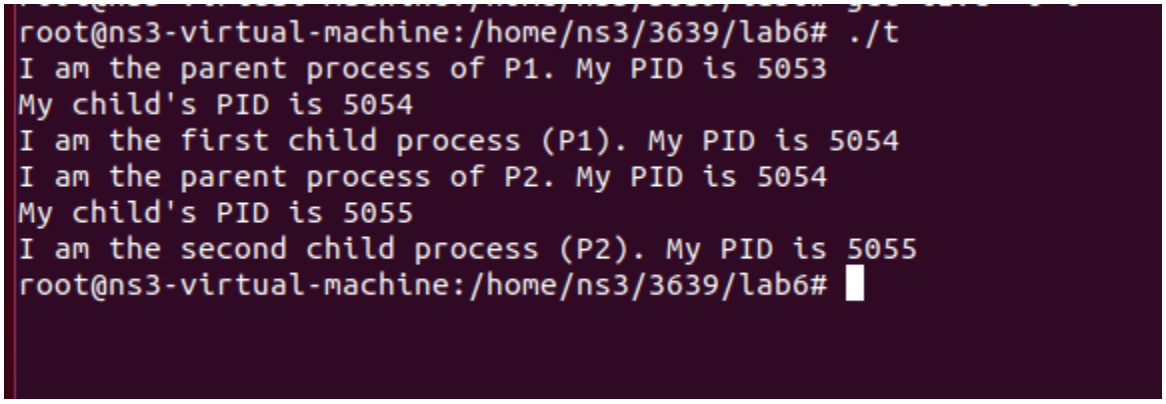
T-1

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

int main() {
    pid_t pid1, pid2;


    pid1 = fork();

    if (pid1 < 0) {

        perror("Failed to create first child process");
        exit(1);
    } else if (pid1 == 0) {

        printf("I am the first child process (P1). My PID is %d\n", getpid());
        exit(0);
    } else {

        pid2 = fork();

        if (pid2 < 0) {

            perror("Failed to create second child process");
```

```
        exit(1);
    } else if (pid2 == 0) {

        printf("I am the second child process (P2). My PID is %d\n", getpid());
        exit(0);
    } else {

        printf("I am the parent process. My PID is %d\n", getpid());
        printf("My first child's PID is %d\n", pid1);
        printf("My second child's PID is %d\n", pid2);
    }
}
```



T-2



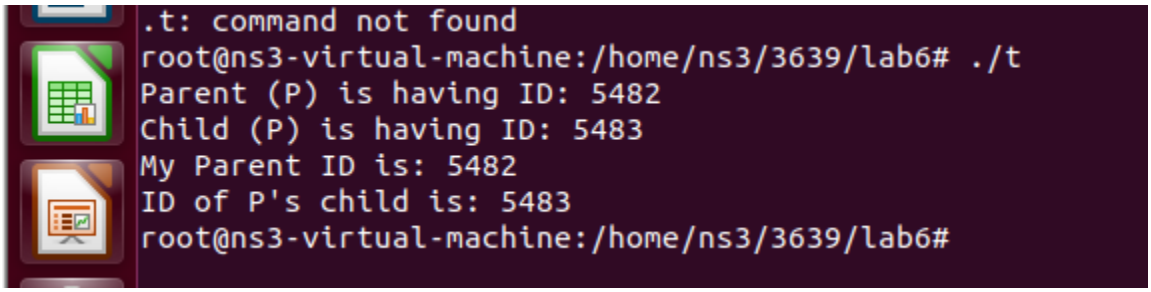

```
#include <stdio.h>
#include <stdlib.h>
```

```c
#include <unistd.h>
#include<sys/wait.h>

int main() {
    pid_t pid1, pid2;


    pid1 = fork();

    if (pid1 < 0) {

        perror("Failed to create first child process");
        exit(1);
    } else if (pid1 == 0) {

        printf("I am the first child process (P1). My PID is %d\n", getpid());
          pid2 = fork();

        if (pid2 < 0) {

            perror("Failed to create second child P2");
            exit(1);
        } else if (pid2 == 0) {

            printf("I am the second child process (P2). My PID is %d\n", getpid());
            exit(0);
        } else {

            printf("I am the parent process of P2. My PID is %d\n", getpid());
            printf("My child's PID is %d\n", pid2);

        }
        exit(0);
    } else {

       printf("I am the parent process of P1. My PID is %d\n", getpid());
            printf("My child's PID is %d\n", pid1);
    }


    if (pid1 > 0 && pid2 > 0) {
        wait(NULL);
        wait(NULL);
    }
```

```
    return 0;
}
```

T-3

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

int main() {
    pid_t pid;


    pid = fork();

    if (pid < 0) {

        perror("Failed to create first child process");
        exit(1);
    } else if (pid == 0) {

        printf("Child (P) is having ID: %d\n", getpid());

        printf("My Parent ID is: %d\n", getppid());

        exit(0);
    } else {

       printf("Parent (P) is having ID: %d\n", getpid());
wait(NULL);
        printf("ID of P's child is: %d\n", pid);
    }




    return 0;
}
```



T-4

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

int main()
{
    pid_t pid;
    const char *filename = "Relation.txt";

    pid = fork();

    if (pid < 0)
    {

        perror("Fork failed");
        exit(1);
    }
    else if (pid == 0)
    {

        int fd = open(filename, O_CREAT | O_WRONLY | O_TRUNC, 0644);
        if (fd < 0)
        {
            perror("Failed to create file");
            exit(1);
```

```
        }

        close(fd);
        exit(0);
    }
    else
    {

        wait(NULL);

        int fd = open(filename, O_WRONLY | O_APPEND);
        if (fd < 0)
        {
            perror("Failed to open file");
            exit(1);
        }

        char buffer[256];
        printf("Enter some content to write into Relation.txt: ");
        scanf("%255s", buffer);

        write(fd, buffer, sizeof(buffer));

        close(fd);
    }

    return 0;
}
```

```
root@ns3-virtual-machine:/home/ns3/3639/lab6# ./t
Enter some content to write into Relation.txt: New Data written by Parent
root@ns3-virtual-machine:/home/ns3/3639/lab6# ls
Relation.txt  t  t1.c
root@ns3-virtual-machine:/home/ns3/3639/lab6#
```

T-5

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

```c
#include <sys/types.h>
#include <sys/wait.h>

int main()
{
    pid_t pid1, pid2;

    pid1 = fork();

    if (pid1 < 0)
    {
        perror("Failed to create first child process");
        exit(1);
    }
    else if (pid1 == 0)
    {

        printf("I am the first child process (PID: %d)\n", getpid());
        sleep(2);
        printf("First child process (PID: %d) is exiting.\n", getpid());
        exit(0);
    }
    else
    {

        pid2 = fork();

        if (pid2 < 0)
        {
            perror("Failed to create second child process");
            exit(1);
        }
        else if (pid2 == 0)
        {

            printf("I am the second child process (PID: %d)\n", getpid());
            sleep(3);
            printf("Second child process (PID: %d) is exiting.\n",
getpid());
            exit(0);
```

```
        }
        else
        {

            printf("I am the parent process (PID: %d)\n", getpid());
            printf("Waiting for both child processes to finish...\n");

            waitpid(pid1, NULL, 0);

            waitpid(pid2, NULL, 0);

            printf("Both child processes have finished. Parent
exiting.\n");
        }
    }

    return 0;
}
```

```
root@ns3-virtual-machine:/home/ns3/3639/lab6# ./t
I am the parent process (PID: 5509)
Waiting for both child processes to finish...
I am the first child process (PID: 5510)
I am the second child process (PID: 5511)
First child process (PID: 5510) is exiting.
Second child process (PID: 5511) is exiting.
Both child processes have finished. Parent exiting.
root@ns3-virtual-machine:/home/ns3/3639/lab6#
```

T-6

```
root@ns3-virtual-machine:/home/ns3/3639/lab6# ./t
I am the first child process (C1). My PID is 5569, Parent PID is 5568
I am the second child process (C2). My PID is 5570, Parent PID is 5568
I am the parent process. My PID is 5568. I am exiting now.
root@ns3-virtual-machine:/home/ns3/3639/lab6# C1 is exiting. My PID is 5569
C2 is exiting. My PID is 5570
```

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

```c
#include <sys/types.h>
#include <sys/wait.h>

int main()
{
    pid_t pid1, pid2;

    pid1 = fork();

    if (pid1 < 0)
    {
        perror("Failed to create first child process (C1)");
        exit(1);
    }
    else if (pid1 == 0)
    {

        printf("I am the first child process (C1). My PID is %d, Parent
PID is %d\n", getpid(), getppid());
        sleep(5);
        printf("C1 is exiting. My PID is %d\n", getpid());
        exit(0);
    }
    else
    {

        pid2 = fork();

        if (pid2 < 0)
        {
            perror("Failed to create second child process (C2)");
            exit(1);
        }
        else if (pid2 == 0)
        {

            printf("I am the second child process (C2). My PID is %d,
Parent PID is %d\n", getpid(), getppid());
            sleep(10);
            printf("C2 is exiting. My PID is %d\n", getpid());
```

```
            exit(0);
        }
        else
        {

            printf("I am the parent process. My PID is %d. I am exiting
now.\n", getpid());
            exit(0);
        }
    }

    return 0;
}
```

T-7

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

int main()
{
    pid_t pid1, pid2;

    pid1 = fork();

    if (pid1 < 0)
    {
        perror("Failed to create first child process (C1)");
        exit(1);
    }
    else if (pid1 == 0)
    {

        printf("I am the first child process (C1). My PID is %d, Parent
PID is %d\n", getpid(), getppid());
```

```
        exit(0);
    }
    else
    {

        pid2 = fork();

        if (pid2 < 0)
        {
            perror("Failed to create second child process (C2)");
            exit(1);
        }
        else if (pid2 == 0)
        {

            printf("I am the second child process (C2). My PID is %d,
Parent PID is %d\n", getpid(), getppid());
            sleep(10);
            exit(0);
        }
        else
        {

            printf("I am the parent process. My PID is %d. I will exit
now, leaving C1 as a zombie and C2 as an orphan.\n", getpid());
            sleep(2);
            exit(0);
        }
    }

    return 0;
}
```

```
I am the parent process. My PID is 5638. I will exit now, leaving C1 as a zombie and C2 as an orphan.
I am the first child process (C1). My PID is 5639, Parent PID is 5638
I am the second child process (C2). My PID is 5640, Parent PID is 5638
root@ns3-virtual-machine:/home/ns3/3639/lab6#
```

T-8

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

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

void calc_fact(int number)
{
    long long factorial = 1;
    for (int i = 1; i <= number; ++i)
    {
        factorial *= i;
    }
    printf("Factorial of %d is %lld\n", number, factorial);
}

void calc_fibonacchi(int steps)
{
    int t1 = 0, t2 = 1, nextTerm;

    printf("Fibonacci Series up to %d steps: ", steps);
    for (int i = 1; i <= steps; ++i)
    {
        printf("%d ", t1);
        nextTerm = t1 + t2;
        t1 = t2;
        t2 = nextTerm;
    }
    printf("\n");
}

int main()
{
    int number, steps;
    pid_t pid;

    printf("Enter a number for factorial calculation: ");
    scanf("%d", &number);
    printf("Enter the number of steps for Fibonacci series: ");
    scanf("%d", &steps);
```

```
    pid = fork();

    if (pid < 0)
    {

        perror("Fork failed");
        exit(1);
    }
    else if (pid == 0)
    {

        calc_fact(number);
        exit(0);
    }
    else
    {

        calc_fibonacchi(steps);

        wait(NULL);
        printf("Child process completed.\n");
    }

    return 0;
}
```

```
root@ns3-virtual-machine:/home/ns3/3639/lab6# ./t
Enter a number for factorial calculation: 8
Enter the number of steps for Fibonacci series: 6
Fibonacci Series up to 6 steps: 0 1 1 2 3 5
Factorial of 8 is 40320
Child process completed.
root@ns3-virtual-machine:/home/ns3/3639/lab6#
```