OS Lab 4

22F-3168

Mariam Fatima

Task 1:

```
#include<stdio.h>
#include<sys/types.h>
#include<unistd.h>
int main()
{
pid_t pid1, pid2;
int num1=5,num2=5;
//First fork to creat p1
pid1 = fork();
if (pid1 == 0)
{
        printf("this is child process p1 with pid: %d\n", getpid());
        int sum = num1+num2;
        printf("Sum of the two numbers is: %d\n", sum);
}
        else{
                //Parent process P forks again to create P2
                pid2= fork();
                if(pid2==0)
                {
                        printf("This is Child process P with pid: %d\n", getpid());
                        int prd = num1*num2;
                        printf("Product of the two numbers is: %d\n",prd);
```

```
                }
                else
                {
                        printf("This is parent Process P wit Pid: %d\n",getpid());
                }
        }
}
```

task1.c (~/) - gedit

Open | task1.c ~/ | Save

```c
#include<stdio.h>
#include<sys/types.h>
#include<unistd.h>
int main()
{
pid_t pid1, pid2;
int num1=5,num2=5;
//First fork to creat p1
pid1 = fork();
if (pid1 == 0)
{
        printf("this is child process p1 with pid: %d\n", getpid());
        int sum = num1+num2;
        printf("Sum of the two numbers is: %d\n", sum);
}
        else{
                //Parent process P forks again to create P2
                pid2= fork();
                if(pid2==0)
                {
                        printf("This is Child process P with pid: %d\n", getpid());
                        int prd = num1*num2;
                        printf("Product of the two numbers is: %d\n",prd);
                }
                else
                {
                        printf("This is parent Process P wit Pid: %d\n",getpid());
                }
        }
}
```

C | Tab Width: 8 | Ln 1, Col 1 | INS

```
ns3@ns3-virtual-machine:~$ nano task1.c
ns3@ns3-virtual-machine:~$ gcc -o q1 task1.c
ns3@ns3-virtual-machine:~$ ./q1
This is parent Process P wit Pid: 13222
ns3@ns3-virtual-machine:~$ This is Child process P with pid: 13224
Product of the two numbers is: 25
this is child process p1 with pid: 13223
Sum of the two numbers is: 10
```

Task2:

#include<stdio.h>

#include<sys/types.h>

```
#include<unistd.h>


int main() {

pid_t p, p1;

p = fork();

if (p == 0) {

   printf("I am the P1 child Process %d and my parent id %d\n", getpid(), getppid());


   p1 = fork();

   if (p1 == 0) {

      printf("I am the P2 child Process %d and my parent id %d\n", getpid(), getppid());

      }

   }

         else if (p > 0) {

         printf("I am the Parent Process P with PID %d\n", getpid());

   }

}
```

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

int main() {
pid_t p, p1;
p = fork();
if (p == 0) {
    printf("I am the P1 child Process %d and my parent id %d\n", getpid(), getppid());

    p1 = fork();
    if (p1 == 0) {
        printf("I am the P2 child Process %d and my parent id %d\n", getpid(), getppid());
        }
    }
        else if (p > 0) {
                printf("I am the Parent Process P with PID %d\n", getpid());
    }
}
```

```
ns3@ns3-virtual-machine:~$ gcc -o q2 task2.c
ns3@ns3-virtual-machine:~$ ./q2
I am the Parent Process P with PID 10990
ns3@ns3-virtual-machine:~$ I am the P1 child Process 10991 and my parent id 1544
I am the P2 child Process 10992 and my parent id 10991
```

Task 3:

```
#include<stdio.h>

#include <unistd.h>


int main() {

   printf("This is the initial program.\n");


   execl("./q3","q3",(char *)NULL);


}
```

```
#include <stdio.h>


int main() {

   int a = 5, b = 3;

   printf("This is the multiplication program.\n");

   printf("Multiplication of %d and %d is %d\n", a, b, a * b);

   return 0;

}
```

```
ns3@ns3-virtual-machine:~$ gcc -o q3_i task3_initial.c
ns3@ns3-virtual-machine:~$ gcc -o q3 task3.c
ns3@ns3-virtual-machine:~$ ./q3_i
This is the initial program.
This is the multiplication program.
Multiplication of 5 and 3 is 15
ns3@ns3-virtual-machine:~$
```

Task 4:

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

int main() {
pid_t pid1;
pid1 = fork();

if (pid == 0) {
   // Child process
   printf("C) Child is having ID %d\n", getpid());        // C
   printf("D) My Parent ID is %d\n", getppid());          // D
} else {
   // Parent process
   printf("A) Parent (P) is having ID %d\n", getpid());    // A

   // Wait for the child process to complete
   wait(NULL);

   printf("B) ID of P's Child is %d\n", pid);             // B
}

}
```

task4.c (~/) - gedit

Open | task4.c ~/ | Save

task3.c × | task4.c ×

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

int main() {
pid_t pid1;
pid1 = fork();

if (pid1 == 0) {
    // Child process
    printf("C) Child is having ID %d\n", getpid());          // C
    printf("D) My Parent ID is %d\n", getppid());            // D
} else {
    // Parent process
    printf("A) Parent (P) is having ID %d\n", getpid());     // A

    // Wait for the child process to complete
    wait(NULL);

    printf("B) ID of P's Child is %d\n", pid1);               // B
}

}
```

```
ns3@ns3-virtual-machine:~$ gcc -o q4 task4.c
ns3@ns3-virtual-machine:~$ ./q4
A) Parent (P) is having ID 11736
C) Child is having ID 11737
D) My Parent ID is 11736
B) ID of P's Child is 11737
ns3@ns3-virtual-machine:~$
```