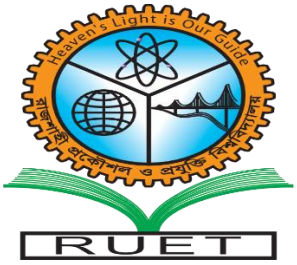
Heaven's Light is Our Guide



Computer Science And Engineering

Rajshahi University of Engineering and Technology

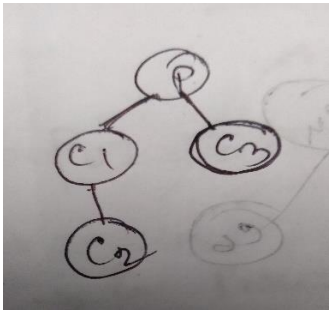
Course No: CSE3202

Course Title: Operating system

Date of Submission: 12-12-22

Submitted To	Submitted By
Mohiuddin Ahmed Lecturer, Department Of Computer Science And Engineering Rajshahi University Of Engineering And Technology	Nazmul Haque Roll: 1803109 Section: B Department Of Computer Science And Engineering Rajshahi University Of Engineering And Technology

Problem-1: Basic fork() operation and verify using child and parent id.



Theory:

fork() function create a duplicate under the portion of this function. if we print 'hello world' under a fork() function then first 'hello world' will be print for it's main program and another 'hello world' will be print for duplicate portion of main code.

1. Code:

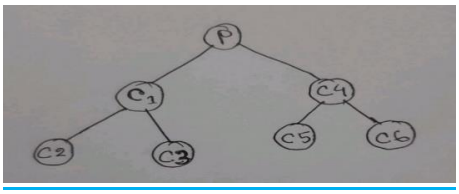
```
nazmul1803109@DESKTOP-VMBU6LB: /mnt/d/Study/3-2 Semister/CSE-3202/Lab-4(28-11-22)
  GNU nano 6.2                                                    fork.c
#include<stdio.h>
#include<unistd.h>
#include<sys/types.h>
int main()
{
fork();
fork();
printf("[son] pid %d from [parent] pid %d\n",getpid(),getppid());
printf("hello world\n");
return 0;
}
```

Input and output:

```
nazmul1803109@DESKTOP-VMBU6LB:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-4(28-11-22)$ nano fork.c
nazmul1803109@DESKTOP-VMBU6LB:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-4(28-11-22)$ gcc fork.c
nazmul1803109@DESKTOP-VMBU6LB:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-4(28-11-22)$ ./a.out
[son] pid 189 from [parent] pid 9
[son] pid 190 from [parent] pid 189
hello world
[son] pid 191 from [parent] pid 190
hello world
[son] pid 192 from [parent] pid 189
hello world
hello world
```

Problem-2:

implement this problem using fork() function and verify using child and parent id.



Theory:

Using getpid() function we can find out the id of child process and id of the parent process can be find out using getppid() function. To run program in backend I use ./a.out & command. Using pstree -p -s $BASHPID command we can show the tree representation of child parent process id . to show this tree representation we need to use sleep() function.

2. Code:

nazmul1803109@DESKTOP-VMBU6LB: /mnt/d/Study/3-2 Semister/CSE-3202/Lab-4(28-11-22)

GNU nano 6.2

```
#include<stdio.h>
#include<unistd.h>
#include<sys/types.h>
int main()
{
int n1=fork();
if(n1>0)
{
 printf("[parent] pid %d\n",getpid());
 sleep(20);
 int n2=fork();
 if(n2==0)
  {
    printf("[son] pid %d from [parent] pid %d \n",getpid(),getppid());
    sleep(20);
   int n5=fork();
    if(n5>0)
     {
       int n6=fork();
       if(n6==0)
        {
          printf("[son] pid %d from [parent] pid %d \n",getpid(),getppid());
          sleep(20);
        }
        sleep(20);

     }
    else if(n5==0)
     {
       printf("[son] pid %d form [parent] pid %d \n",getpid(),getppid());
       sleep(20);
     }
     sleep(20);
  }
}
else if(n1==0)
{
  printf("[son] pid %d form [parent] pid %d \n",getpid(),getppid());
  sleep(20);
 int n3=fork();
```

```c
else if(n1==0)
{
  printf("[son] pid %d form [parent] pid %d \n",getpid(),getppid());
  sleep(20);
 int n3=fork();
  if(n3>0)
   {
     int n4=fork();
     if(n4==0)
      {
        printf("[son] pid %d from [parent] pid %d \n",getpid(),getppid());
        sleep(20);
      }
      sleep(20);
   }
  else if(n3==0)
    {
       printf("[son] pid %d from [parent] pid %d \n",getpid(),getppid());
       sleep(20);
    }
     sleep(20);

}

return 0;
}
```

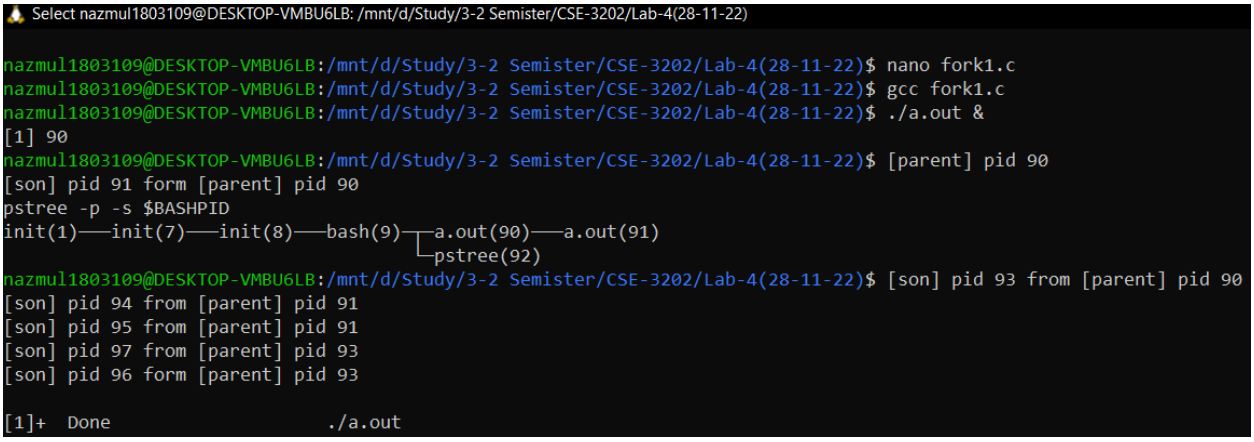
Input And Output:

Input output when taking sleep with 10 s.

```
Select nazmul1803109@DESKTOP-VMBU6LB: /mnt/d/Study/3-2 Semister/CSE-3202/Lab-4(28-11-22)
nazmul1803109@DESKTOP-VMBU6LB:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-4(28-11-22)$ ./a.out &
[1] 76
nazmul1803109@DESKTOP-VMBU6LB:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-4(28-11-22)$ [parent] pid 76
[son] pid 77 form [parent] pid 76
pstree -p -s $BA[son] pid 78 from [parent] pid 76
[son] pid 79 from [parent] pid 77
[son] pid 80 from [parent] pid 77

init(1)─┬─init(4)───{init}(5)
        ├─init(7)───init(8)─┬─a.out(77)─┬─a.out(79)
        │                   │           └─a.out(80)
        │                   ├─a.out(78)
        │                   └─bash(9)───pstree(81)
        └─{init}(6)
[1]+  Done                    ./a.out
nazmul1803109@DESKTOP-VMBU6LB:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-4(28-11-22)$ [son] pid 82 form [parent] pid 78
[son] pid 83 from [parent] pid 78
```

Input output when taking sleep with 20 s.



Discussion:

The above two codes run successfully except tree representation. Though, I use sleep function with 10s at first attempt then at second attempt take sleep function with 20s.But don’t show proper tree representation of child ,parent process id. Although show correct child, parent process id.