OS LAB EXPT-3

Roehit Ranganathan | RA1911033010017 | L2 cse-swe

1. Process Creation using C.

Procedure:

Step 1: Create a child process using fork () command

Display the child process content

Step 2: Display the content from current process

Step 3: Stop the process

Expected Output:

SRMIST

SRMIST

```
roehit@LAPTOP-0SIPK43K:~$ nano one.c
roehit@LAPTOP-0SIPK43K:~$ gcc one.c
roehit@LAPTOP-0SIPK43K:~$ .a/.out
-bash: .a/.out: No such file or directory
roehit@LAPTOP-0SIPK43K:~$ ./a.out
srmist
srmist
roehit@LAPTOP-0SIPK43K:~$
```

```
#include <stdio.h>
#include <unistd.h>
int main()
{
fork();
printf("srmist\n");
return 0;
}
```

2. Display process details using C.

Procedure:

Step 1: Create a process

Step 2: Create a parent process

Step 3: Get the process ID

Step 4: Display the process ID

Step 5: Get the Parent process ID

Step 6: Display the Parent process ID

Step 7: Stop the process

```
roehit@LAPTOP-0SIPK43K:~$ nano three.c
roehit@LAPTOP-0SIPK43K:~$ gcc three.c
roehit@LAPTOP-0SIPK43K:~$ ./a.out
Process id id 81
Parent Process id is 8roehit@LAPTOP-0SIPK43K:~$
```

```
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>
int main()
{
pid_t process_id;
pid_t p_process_id;
process_id=getpid();
p_process_id=getppid();
printf("Process id id %d",process_id);
printf("\nParent Process id is %d",p_process_id);
return 0;
}
```

3. Different process Execution for parent and child process using C.

Procedure:

Step 1: Create a child process

Step 2: If the process is called by child

Step 3: Execute the child process

Step 4: else

Step 5: Execute the parent process

Step 6: End if

Step 7: stop the process

```
Parent Process id is 8roehit@LAPTOP-0SIPK43K:~$ nano two.c
roehit@LAPTOP-0SIPK43K:~$ gcc two.c
roehit@LAPTOP-0SIPK43K:~$ ./a.out
In parent process
In child process
Child Process ID 88
Child Process ID 89
Parent Process ID:8
Parent Process ID:88
```

```c
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <sys/wait.h>
int main(){
int id=fork();
if(id==0)
{
printf("In child process \nChild Process ID %d\nParent Process ID:%d\n ",getpid(),getppid());
}
else
{
printf("In parent process \nChild Process ID %d\nParent Process ID:%d\n ",getpid(),getppid());
}
return 0;
}
```

4. Clone process execution using C.

Procedure:

Step 1: Create a clone process

Step 2: If the process is called by clone

Step 3: Execute the clone process

Step 4: else

Step 5: Execute the parent process

Step 6: End if

Step 7: stop the process

```
roehit@LAPTOP-0SIPK43K:~$ nano four.c
roehit@LAPTOP-0SIPK43K:~$ gcc four.c
roehit@LAPTOP-0SIPK43K:~$ ./a.out
Child process started
value of n: 10
Now i am coming back to parent process
value of n: -2027119648
```

```c
#include <stdio.h>
#include <unistd.h>
int main()
{
int n=10;
pid_t pid = vfork();
if (pid ==0)
{
printf("Child process started\n");
}
else
{
printf("Now i am coming back to parent process\n");
}
printf("value of n: %d\n",n);
return 0;
}
```