**Q1.**

#include<stdio.h>

#include<unistd.h>

#include<sys/wait.h>

int main()

{

pid\_t pid;

pid = fork();

if (pid<0)

printf("fork sys failed.");

else if (pid == 0)

{

printf("child process: %d\n",getpid());

printf("parent process: %d\n",getppid());

}

else

{

printf("parent process: %d\n",getpid());

printf("process (main): %d\n",getppid());

}

}

1. **State of both processes** :

parent process: 463

process (main): 296

child process: 464

parent process: 463

Both Parent and Child Process state is R (Running)

0 R 1000 463 296 -

1 R 1000 464 463 -

1. **State of both processes** :

Parent Process state : R (Running)

Child Process state : Z (Zombie)

0 R 1000 463 296

1 Z 1000 464 463

1. **State of both processes** :

parent process: 494

process (main): 296

Parent Process state : Completed

Child Process state : R (Orphan) … Points to another Parent process as its calling Parent\_process got terminated first while it is in running state.

child process: 495

parent process: 494

1 R 1000 495 295

1. **State of both processes** :

parent process: 506

Parent Process state : S (Blocked) , as Wait(NULL); system call will execute child process first by blocking the Parent Process.

Child Process state : R (Running)

child process: 507

parent process -> id of child: 506

0 S 1000 506 296

1 R 1000 507 506

1. **State of both processes** :

parent process: 506

child process: 507

Parent Process state : R (Running) , as the child process which blocks the Parent\_process got executed and terminated that gives signal to Parent Process for switching its state 🡪 Running.

Child Process state : Completed

parent process -> id of child: 506

root process (main): 296

0 R 1000 506 296

**Q3.**

#include<stdio.h>

#include<unistd.h>

#include<sys/wait.h>

int main()

{

if (fork()==0)

{

printf("Child 1 %d\n",getpid());

printf("Parent id of child 1: %d \n", getppid());

execlp("/home/student/2241016519\_Unix/Q3\_script.sh","Q3\_script.sh",NULL);

}

else if (fork()==0)

{

sleep(10);

printf("Child 2 %d\n",getpid());

printf("Parent id of child 2: %d \n", getppid());

execlp("/home/student/2241016519\_Unix/Q3\_script1.sh","Q3\_script1.sh",NULL);

}

else if (fork()==0)

{

sleep(15);

printf("Child 3 %d\n",getpid());

printf("Parent id of child 3: %d \n", getppid());

execlp("/home/student/2241016519\_Unix/Q3\_script2.sh","Q3\_script2.sh",NULL);

}

else

{

sleep(25);

printf("Parent process id: %d \n", getpid());

}

}

**Q3\_script.sh :**

#! /bin/bash

echo "Replacing Child 1 with Q3.scipt"

cp f1.txt f2.txt

echo "Content of f1.txt is copied to f2.txt"

**Q3\_script1.sh :**

#! /bin/bash

echo "Child 2 displaying content of the f2.txt"

cat f2.txt

**Q3\_script2.sh :**

#! /bin/bash

echo "Child 3 displaying content of the f2.txt in Sorted Order"

sort -nr f2.txt

**Output :**

srikant@SRIKANTs-PC:~/DOS\_2241016519/ass4$ gcc Q3.c

srikant@SRIKANTs-PC:~/DOS\_2241016519/ass4$ ./a.out

Child 1 1075

Parent id of child 1: 1074

Replacing Child 1 with Q3.scipt

Content of f1.txt is copied to f2.txt

Child 2 1076

Parent id of child 2: 1074

Child 2 displaying content of the f2.txt

1

2

6

4

12

7

5

19

33

26

Child 3 1077

Parent id of child 3: 1074

Child 3 displaying content of the f2.txt in Sorted Order

33

26

19

12

7

6

5

4

2

1

Parent process id: 1074

srikant@SRIKANTs-PC:~/DOS\_2241016519/ass4$

**Q4.**

#include<stdio.h>

#include<unistd.h>

#include<sys/wait.h>

int main()

{

int a[500];

int n,i;

a[0]=0;

a[1]=1;

if (vfork()==0)

{

printf("Child 1 %d\n",getpid());

printf("Parent id of child 1: %d \n", getppid());

printf("Enter how many numbers: \n");

scanf("%d",&n);

for(i=2;i<n;i++)

{

a[i]=a[i-1]+a[i-2];

}

\_exit(0);

}

else

{

printf("Parent process id: %d \n", getpid());

for(i=0;i<n;i++)

printf(" [%d] ",a[i]);

}

}

**Output :**

srikant@SRIKANTs-PC:~/DOS\_2241016519/ass4$ ./a.out

Child 1 450

Parent id of child 1: 449

Enter how many numbers:

20

Parent process id: 449

[0] [1] [1] [2] [3] [5] [8] [13] [21] [34] [55] [89] [144] [233] [377] [610] [987] [1597] [2584] [4181]