**OPERATING SYSTEM LAB 3**

**Q1.**

PROGRAM:

#include <stdio.h>

#include <unistd.h>

#include <sys/types.h>

#include <stdlib.h>

#include <sys/wait.h>

int main()

{

pid\_t pid;

int status;

pid=fork();

switch(pid)

{

case -1:

printf("Fork failed");

break;

case 0:

for(int i=0;i<5;i++)

{

printf("%d inside child\n", i+1);

sleep(1);

}

exit(0);

default:

wait(&status);

printf("In parent\n");

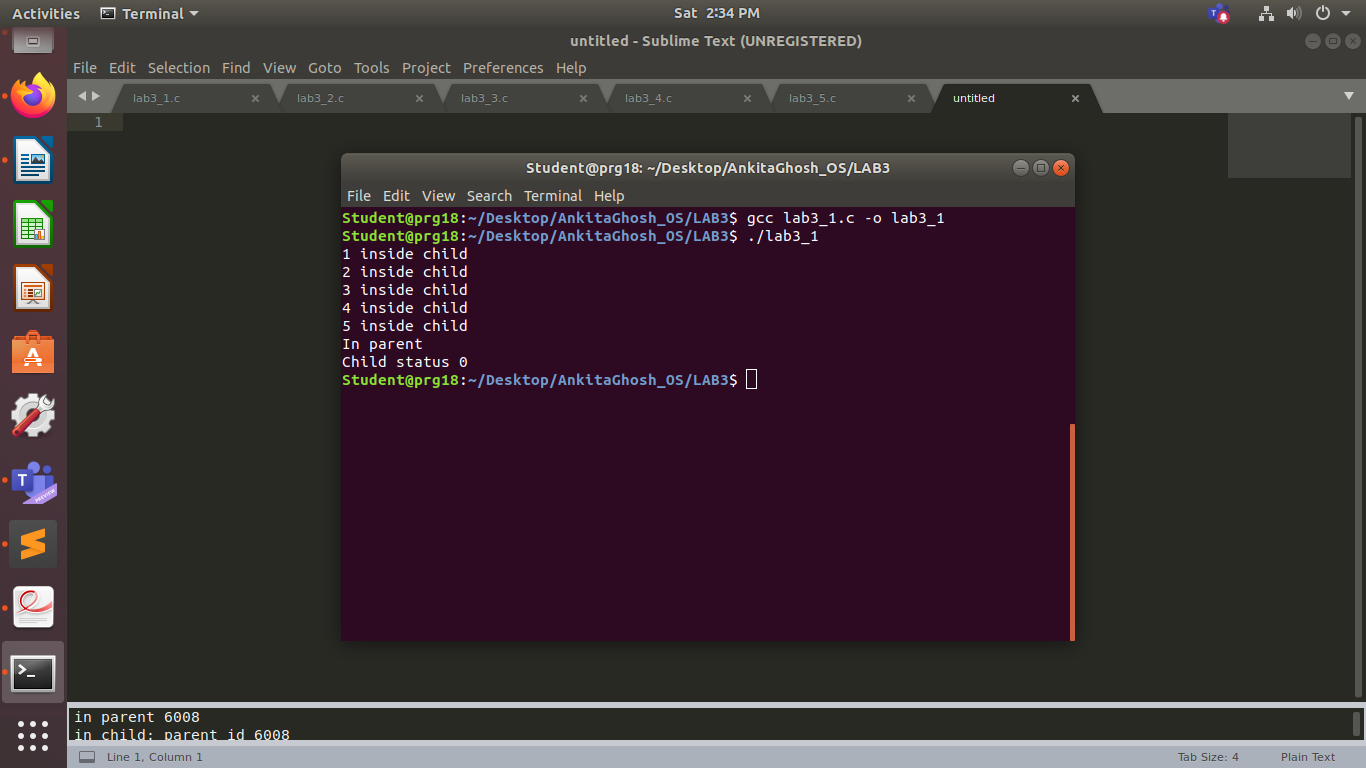
printf("Child status %d\n", status);

}

return 0;

}

OUTPUT:



**Q2.**

PROGRAM:

#include <stdio.h>

#include <unistd.h>

#include <sys/types.h>

#include <stdlib.h>

#include <sys/wait.h>

int main()

{

pid\_t pid;

pid=fork();

switch(pid)

{

case -1:

printf("Fork failed");

break;

case 0:

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

execlp("./lab3\_1","lab3\_1",NULL);

//sleep(1);

default:

wait(NULL);

//sleep(1);

printf("Child complete\n");

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

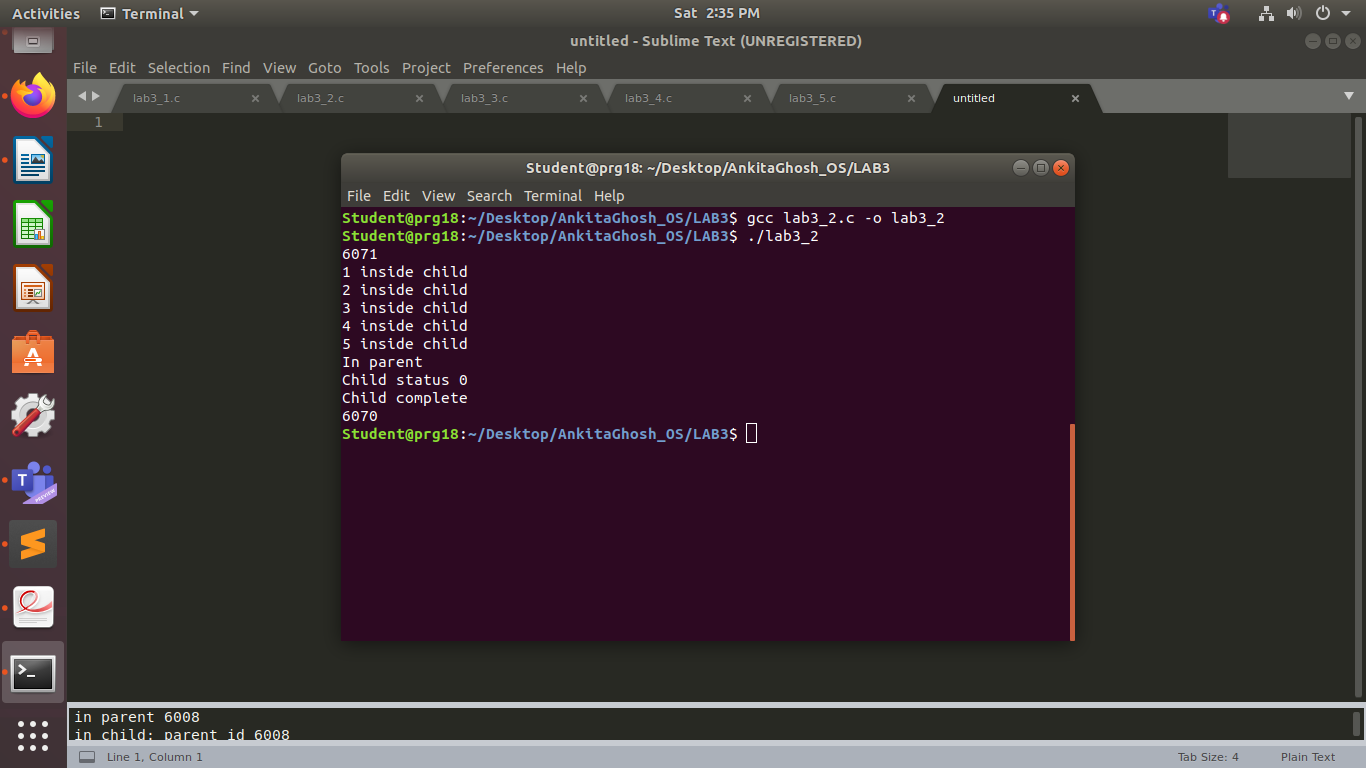
exit(0);

}

return 0;

}

OUTPUT:



**Q3.**

PROGRAM:

#include <stdio.h>

#include <unistd.h>

#include <sys/types.h>

#include <stdlib.h>

#include <sys/wait.h>

int main()

{

pid\_t pid;

pid=fork();

if(pid==-1)

printf("Fork failed");

else if(pid==0)

{

printf("in child process\n");

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

printf("Parent pid: %d\n",getppid());

exit(0);

}

else

{

wait(NULL);

printf("in parent process\n");

printf("Child pid: %d\n",pid);

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

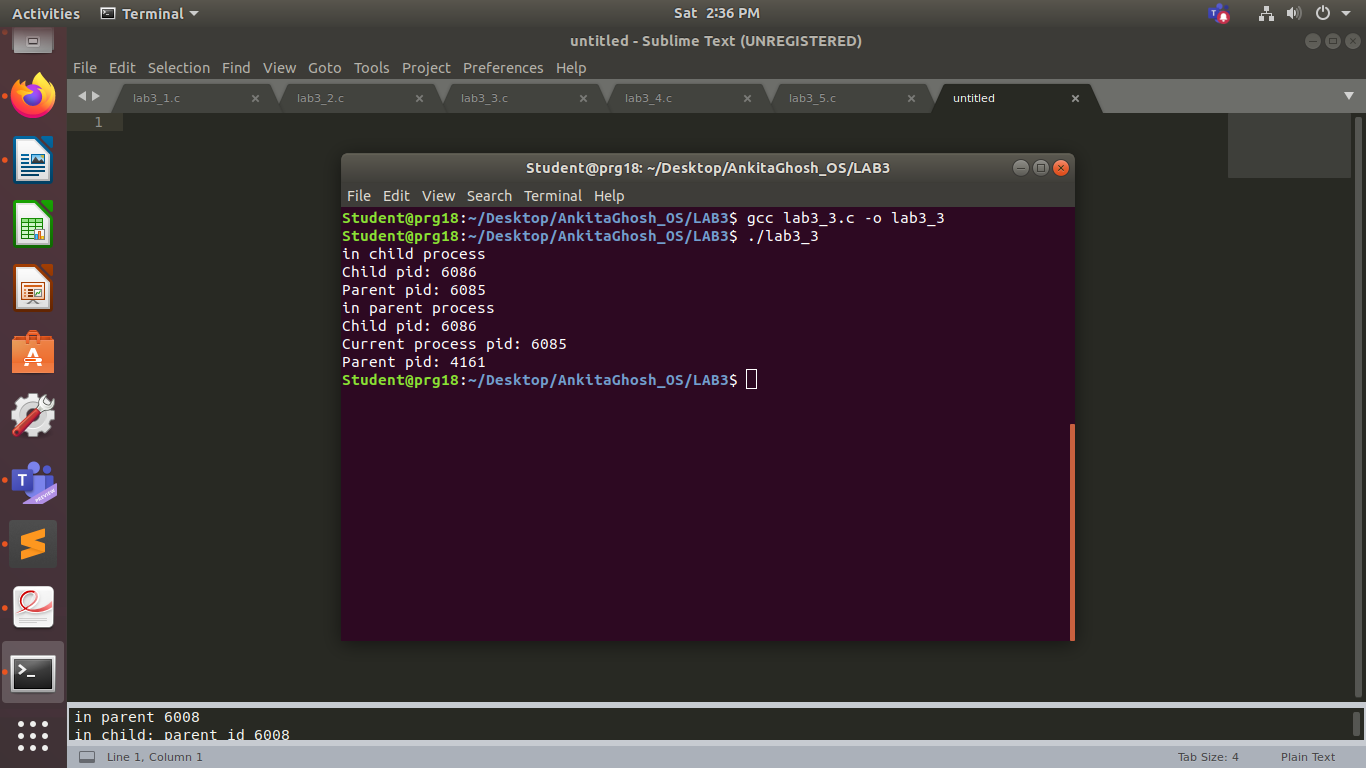
printf("Parent pid: %d\n",getppid());

}

return 0;

}

OUTPUT:



**Q4.**

PROGRAM:

#include <stdio.h>

#include <unistd.h>

#include <sys/types.h>

#include <stdlib.h>

#include <sys/wait.h>

int main()

{

pid\_t pid;

pid = fork();

if(pid==-1)

printf("Fork failed\n");

else if(pid==0)

{

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

exit(0);

}

else

{

sleep(10);

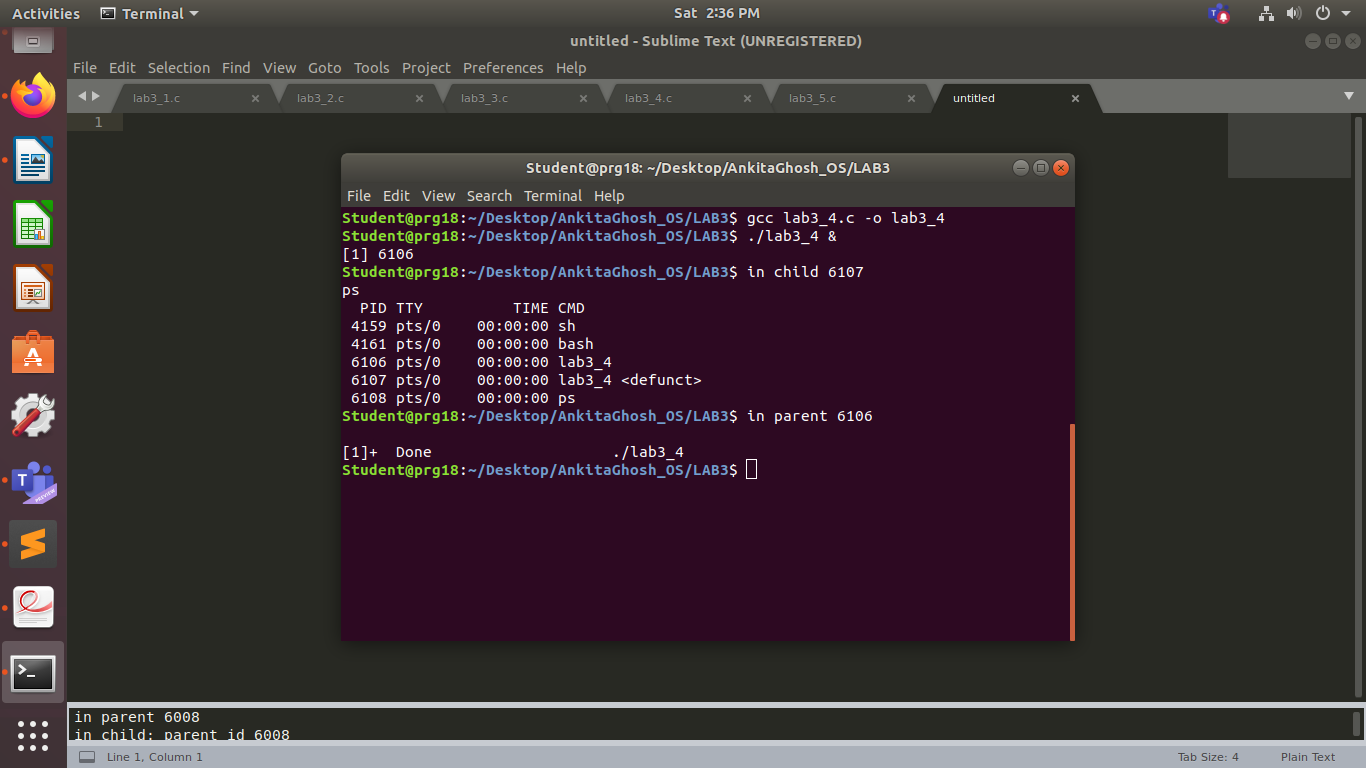
printf("in parent %d\n",getpid());

}

return 0;

}

OUTPUT:



**Q5.**

PROGRAM:

#include <stdio.h>

#include <unistd.h>

#include <sys/types.h>

#include <stdlib.h>

#include <sys/wait.h>

int main()

{

pid\_t pid;

int status;

pid = fork();

if(pid==-1)

printf("Fork failed\n");

else if(pid==0)

{

printf("in child: parent id %d\n",getppid());

sleep(10);

printf("in child: parent id %d\n",getppid());

exit(0);

}

else

{

//wait(&status);

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

//printf("child status %d\n", status);

}

return 0;

}

OUTPUT:

