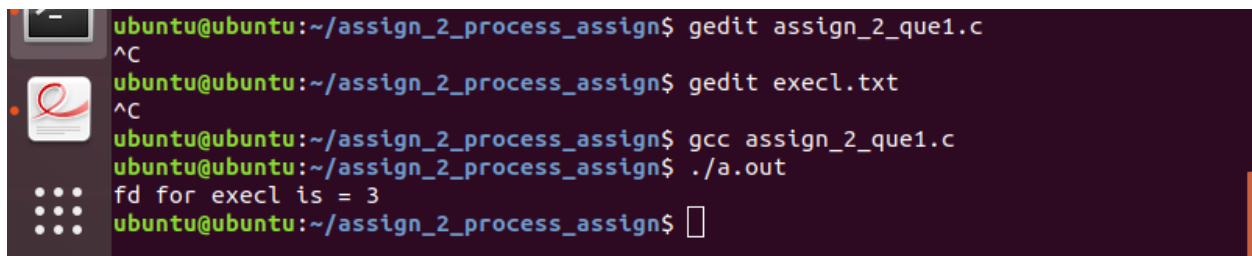


**Khushbu Patel ( 150034 )**

## **ASSIGNMENT 2 ( PROCESS ASSUGNMENT )**

### **QUE : 1**

```
#include<stdio.h>
#include<fcntl.h>
#include<unistd.h>
int main()
{
int fd;
fd=open("execl.txt",O_RDONLY,777);
printf("fd for execl is = %d\n",fd);
close(fd);
return 0;
}
```

A terminal window with a dark purple background. The user is in the directory ~/assign\_2\_process\_assign. They run 'gedit assign\_2\_que1.c' and 'gedit execl.txt'. Then they compile with 'gcc assign\_2\_que1.c' and run './a.out'. The output is 'fd for execl is = 3'.

```
ubuntu@ubuntu:~/assign_2_process_assign$ gedit assign_2_que1.c
^C
ubuntu@ubuntu:~/assign_2_process_assign$ gedit execl.txt
^C
ubuntu@ubuntu:~/assign_2_process_assign$ gcc assign_2_que1.c
ubuntu@ubuntu:~/assign_2_process_assign$ ./a.out
fd for execl is = 3
ubuntu@ubuntu:~/assign_2_process_assign$
```

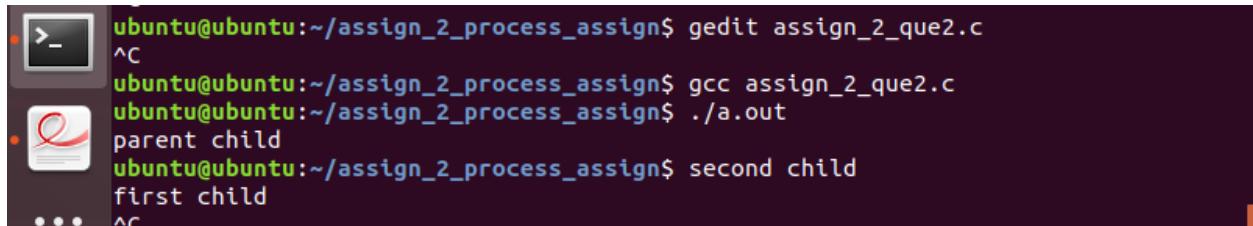
### **QUE : 2**

```
#include<stdio.h>
#include<unistd.h>
#include<stdlib.h>
int main()
{
int pid_1,pid_2;
pid_1=fork();
if(pid_1==0)
{
printf("first child\n");
}
else
{
pid_2=fork();
if(pid_2==0)
{
printf("second child\n");
}
```

```

else
{
printf("parent child\n");
}
return 0;
}

```



```

ubuntu@ubuntu:~/assign_2_process_assign$ gedit assign_2_que2.c
^C
ubuntu@ubuntu:~/assign_2_process_assign$ gcc assign_2_que2.c
ubuntu@ubuntu:~/assign_2_process_assign$ ./a.out
parent child
ubuntu@ubuntu:~/assign_2_process_assign$ second child
first child
^C

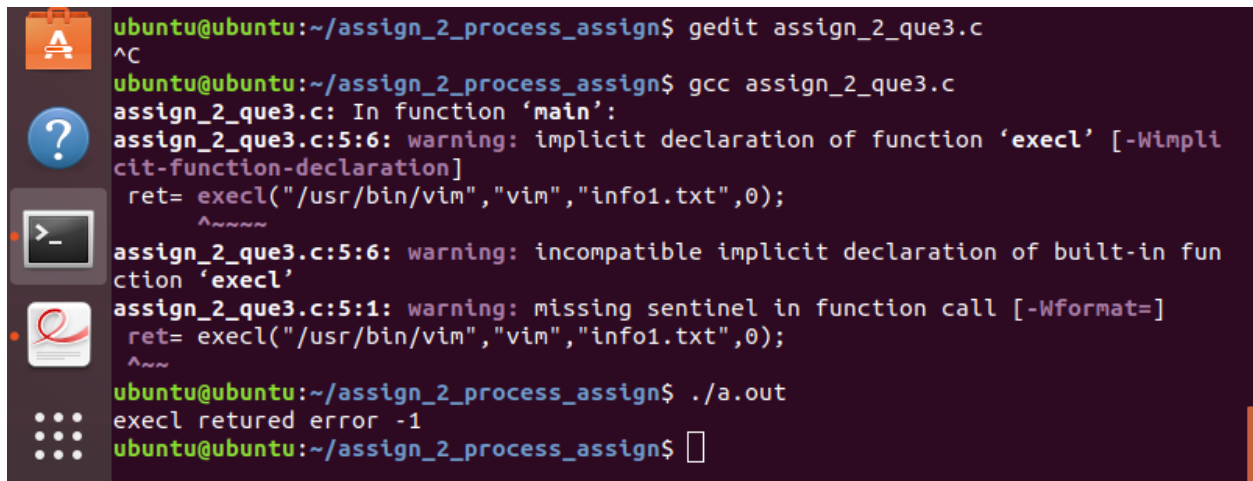
```

### QUE : 3

```

#include<stdio.h>
int main()
{
int ret;
ret= execl("/usr/bin/vim","vim","info1.txt",0);
if(ret==-1)
{
printf("execl retured error %d\n",ret);
}
}

```



```

ubuntu@ubuntu:~/assign_2_process_assign$ gedit assign_2_que3.c
^C
ubuntu@ubuntu:~/assign_2_process_assign$ gcc assign_2_que3.c
assign_2_que3.c: In function 'main':
assign_2_que3.c:5:6: warning: implicit declaration of function 'execl' [-Wimplicit-function-declaration]
    ret= execl("/usr/bin/vim","vim","info1.txt",0);
           ^~~~~
assign_2_que3.c:5:6: warning: incompatible implicit declaration of built-in function 'execl'
assign_2_que3.c:5:1: warning: missing sentinel in function call [-Wformat=]
    ret= execl("/usr/bin/vim","vim","info1.txt",0);
    ~~~~~
ubuntu@ubuntu:~/assign_2_process_assign$ ./a.out
execl retured error -1
ubuntu@ubuntu:~/assign_2_process_assign$ 

```

### QUE : 4

```

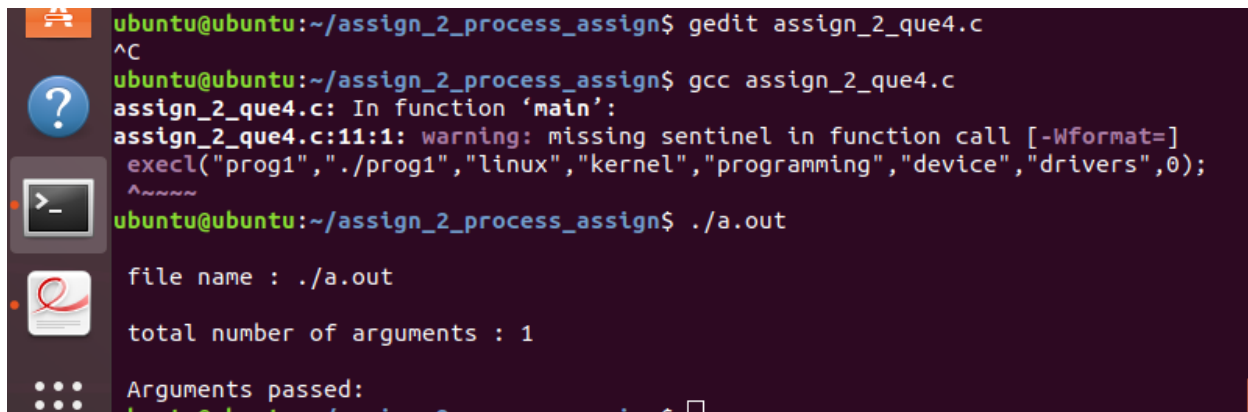
#include<stdio.h>
#include<unistd.h>
#include<stdlib.h>
#include<sys/types.h>
#include<sys/wait.h>

```

```

int main(int argc,char *argv[])
{
int i;
printf("\n file name : %s\n",argv[0]);
printf("\n total number of arguments : %d\n",argc);
execl("prog1","./prog1","linux","kernel","programming","device","drivers",0);
printf("\n Arguments passed: ");
for(i=1;i<argc;i++)
{
printf("%s",argv[i]);}
printf("\n");
return 0;
}

```



```

ubuntu@ubuntu:~/assign_2_process_assign$ gedit assign_2_que4.c
^C
ubuntu@ubuntu:~/assign_2_process_assign$ gcc assign_2_que4.c
assign_2_que4.c: In function 'main':
assign_2_que4.c:11:1: warning: missing sentinel in function call [-Wformat=]
    execl("prog1","./prog1","linux","kernel","programming","device","drivers",0);
    ^~~~~~
ubuntu@ubuntu:~/assign_2_process_assign$ ./a.out

file name : ./a.out

total number of arguments : 1

Arguments passed:

```

## QUE : 5

```

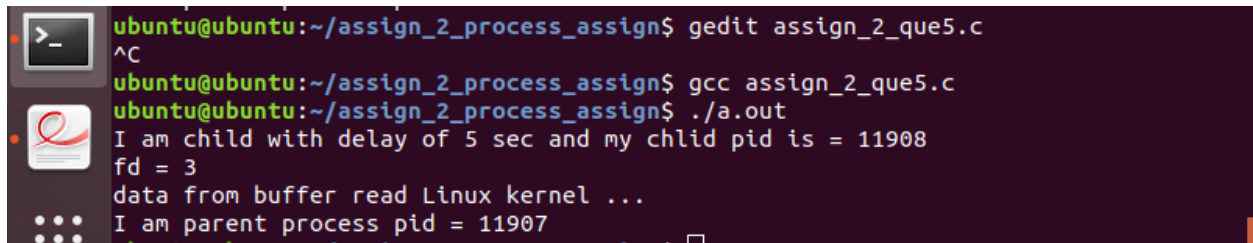
#include<stdio.h>
#include<fcntl.h>
#include<unistd.h>
#include<sys/types.h>
#include<sys/wait.h>
int main()
{
int fd;
int len;
char write_buf[60]="Linux kernel ...";
char read_bf[60];
pid_t pid;
pid=fork();
if(pid==0)
{
sleep(5);
printf("I am child with delay of 5 sec and my chlid pid is = %d\n",getpid());
fd = open("program_5.txt",O_CREAT|O_RDWR,777);

```

```

printf("fd = %d\n",fd);
if(fd>0)
{
len=write(fd,write_buf,60);
}
else
{
printf("Error.....\n");}
//printf("return value from write %d\n",len);
lseek(fd,0,SEEK_SET);
read(fd,read_buf,len);
printf("data from buffer read %s\n",read_buf);
}
else
{
wait(0);
printf("I am parent process pid = %d\n",getpid());
}
close(fd);
return 0;
}

```



```

ubuntu@ubuntu:~/assign_2_process_assign$ gedit assign_2_que5.c
^C
ubuntu@ubuntu:~/assign_2_process_assign$ gcc assign_2_que5.c
ubuntu@ubuntu:~/assign_2_process_assign$ ./a.out
I am child with delay of 5 sec and my chlid pid is = 11908
fd = 3
data from buffer read Linux kernel ...
I am parent process pid = 11907

```

## QUE : 6

```

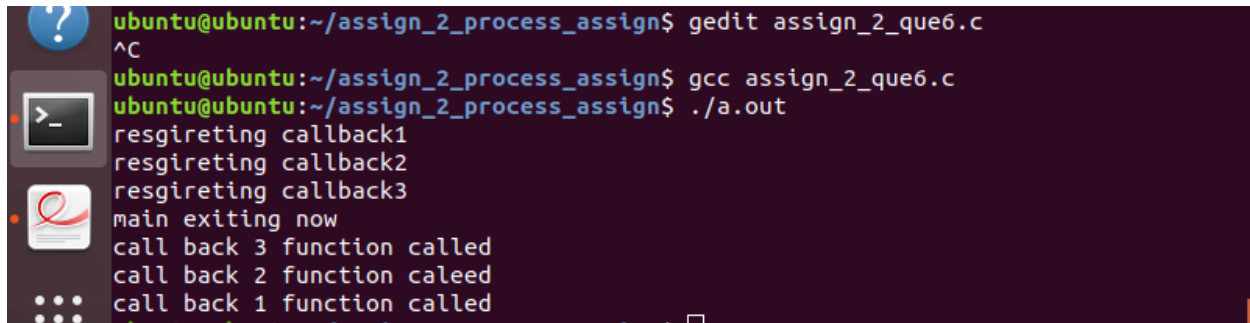
#include<stdio.h>
#include<unistd.h>
#include<stdlib.h>
void callback1(void)
{
printf("call back 1 function called \n");
}
void callback2(void)
{
printf("call back 2 function caleed \n");
}
void callback3(void)
{
printf("call back 3 function called \n");
}
int main()

```

```

{
printf("resgireting callback1\n");
atexit(callback1);
printf("resgireting callback2\n");
atexit(callback2);
printf("resgireting callback3\n");atexit(callback3);
printf("main exiting now\n");
exit(0);
}

```



A terminal window showing the execution of a C program. The user enters 'gedit assign\_2\_que6.c' to open the file, then '^C' to cancel. Next, they enter 'gcc assign\_2\_que6.c' to compile it, and './a.out' to run it. The program's output is displayed: 'resgireting callback1', 'resgireting callback2', 'resgireting callback3', 'main exiting now', 'call back 3 function called', 'call back 2 function caled', and 'call back 1 function called'.

```

ubuntu@ubuntu:~/assign_2_process_assign$ gedit assign_2_que6.c
^C
ubuntu@ubuntu:~/assign_2_process_assign$ gcc assign_2_que6.c
ubuntu@ubuntu:~/assign_2_process_assign$ ./a.out
resgireting callback1
resgireting callback2
resgireting callback3
main exiting now
call back 3 function called
call back 2 function caled
call back 1 function called

```

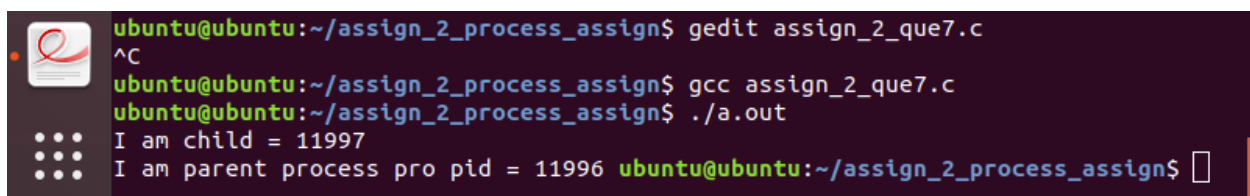
## QUE : 7

```

#include<stdio.h>
#include<unistd.h>
#include<sys/wait.h>
int main()
{
pid_t pid1;
pid1=fork();
if(pid1==0)
{

sleep(5);
printf("I am child = %d \n",getpid());
}
else
{
int pid2;
pid2=wait(0);
printf("I am parent process pro pid = %d ",getpid());
}
}

```



A terminal window showing the execution of a C program. The user enters 'gedit assign\_2\_que7.c' to open the file, then '^C' to cancel. Next, they enter 'gcc assign\_2\_que7.c' to compile it, and './a.out' to run it. The program's output is displayed: 'I am child = 11997' and 'I am parent process pro pid = 11996'.

```

ubuntu@ubuntu:~/assign_2_process_assign$ gedit assign_2_que7.c
^C
ubuntu@ubuntu:~/assign_2_process_assign$ gcc assign_2_que7.c
ubuntu@ubuntu:~/assign_2_process_assign$ ./a.out
I am child = 11997
I am parent process pro pid = 11996 ubuntu@ubuntu:~/assign_2_process_assign$ 

```