Code and Output :-

2. **Write a program for process creation using C**

- Orphan Process

Code:-

```
  GNU nano 7.2
#include <stdio.h>
#include <unistd.h>

int main() {
    if (fork() == 0) {
        sleep(5);
        printf("Child Process\n");
        printf("PID: %d\n", getpid());
        printf("PPID: %d\n", getppid());
    }
    else {
        printf("Parent exiting\n");
    }
    return 0;
}
```

Output:-

```
m309@m309-BY-OEM:~$ nano fork.c
m309@m309-BY-OEM:~$ nano orphan.c
m309@m309-BY-OEM:~$ ./orphan
bash: ./orphan: No such file or directory
m309@m309-BY-OEM:~$ gcc orphan.c -o orphan
m309@m309-BY-OEM:~$ ./orphan
Parent exiting
m309@m309-BY-OEM:~$ Child Process
PID: 6313
PPID: 2095
```

- Zombie Process

Code:-

```
  GNU nano 7.2
#include <stdio.h>
#include <unistd.h>

int main() {
    if (fork() == 0) {
        printf("Child exiting\n");
    }
    else {
        sleep(10);
        printf("Parent sleeping\n");
    }
    return 0;
}
```

Output:-

```
m309@m309-BY-OEM:~$ nano orphan.c
m309@m309-BY-OEM:~$ nano zombie.c
m309@m309-BY-OEM:~$ gcc zombie.c -o zombie
m309@m309-BY-OEM:~$ ./zombie
Child exiting

Parent sleeping
```

4. **Create the process using fork 0 system call**
   - Child Process creation
   - Parent Process creation

Code:-

```
 GNU nano 7.2                                                  fork.c
#include <stdio.h>
#include <unistd.h>

int main() {
    pid_t pid;

    pid = fork();

    if (pid == 0) {
        printf("Child Process\n");
        printf("Child PID: %d\n", getpid());
        printf("Parent PID: %d\n", getppid());
    }
    else {
        printf("Parent Process\n");
        printf("Parent PID: %d\n", getpid());
    }

    return 0;
}
```

Output:-

```
m309@m309-BY-OEM:~$ nano fork.c
m309@m309-BY-OEM:~$ gcc fork.c -o fork
m309@m309-BY-OEM:~$ ./fork
Parent Process
Parent PID: 5245
Child Process
Child PID: 5246
Parent PID: 5245
m309@m309-BY-OEM:~$
```