

# Practical no: 4

## 1.orphan process

```
#include <stdio.h>
#include <unistd.h>

int main() {
    pid_t pid = fork();

    if (pid > 0) {
        printf("Parent Process ID: %d\n", getpid());
        sleep(2);
    } else {
        sleep(5);
        printf("Child Process ID: %d\n", getpid());
        printf("Parent Process ID: %d\n", getppid());
    }
    return 0;
}
```

## 2. Zombie process

```
#include <stdio.h>
#include <unistd.h>

int main() {
    pid_t pid = fork();

    if (pid > 0) {
        printf("Parent Process ID: %d\n", getpid());
        sleep(20);
    }
```

```
    } else {  
        printf("Child Process ID: %d\n", getpid());  
    }  
    return 0;  
}
```

### 3. Fork

```
#include <stdio.h>  
#include <unistd.h>  
  
int main() {  
    pid_t pid = fork();  
  
    if (pid == 0) {  
        printf("Child Process\n");  
        printf("PID: %d\n", getpid());  
        printf("Parent PID: %d\n", getppid());  
    } else {  
        printf("Parent Process\n");  
        printf("PID: %d\n", getpid());  
        printf("Child PID: %d\n", pid);  
    }  
}
```