



OS LAB 5

Ayush Singh

209301163

Demonstrating child process

```
#include<unistd.h>
#include<stdio.h>
#include<sys/types.h>

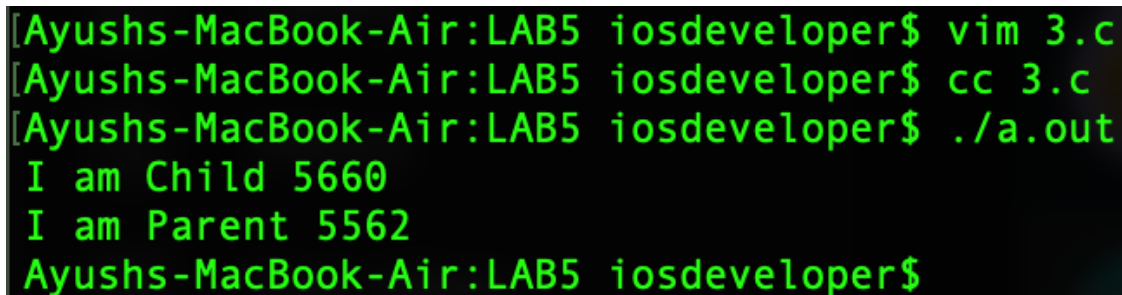
int main() {
    int PID=fork() ;
    if (PID>0){
        printf ("parent %d\n", getppid());
    }
    else if (PID==0) {
        printf("CHILD %d" , getpid());
    }
    else {
        printf ("ERROR") ;
    }
    return 0;
}
```

```
[Ayushs-MacBook-Air:LAB5 iosdeveloper$ ls
[Ayushs-MacBook-Air:LAB5 iosdeveloper$ vim 1.c
[Ayushs-MacBook-Air:LAB5 iosdeveloper$ cc 1.c
[Ayushs-MacBook-Air:LAB5 iosdeveloper$ ./a.out
parent 5562
CHILD 5585Ayushs-MacBook-Air:LAB5 iosdeveloper$
```

Demonstrating orphan process

```
#include<unistd.h>
#include<stdio.h>
#include<sys/types.h>

int main(){
    pid_t p= fork();
    if(p==0){
        sleep(5);
        printf("I am Child : %d\n", getpid());
        printf ("Parent %d\n", getppid());
    }else{
        printf("I am Parent %d\n", getppid());
        printf ("Child : %d\n", getpid());
    }
    return 0;
}
```



A terminal window with a black background and green text. The prompt is [Ayushs-MacBook-Air:LAB5 iosdeveloper\$. The commands and their outputs are: vim 3.c, cc 3.c, ./a.out. The output shows 'I am Child 5660' followed by 'I am Parent 5562' on the next line, indicating the child process became an orphan after its parent finished.

Demonstrating zombie process

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

int main(){
    pid_t PID = fork();
    if(PID>0) {
        sleep(10);
        printf("I am Parent %d\n", getppid());
    }
    else if(PID==0){
```

```

printf("I am Child %d\n" , getpid());
exit(0);
}else{
printf("ERROR");
}
return 0;
}

```

```

Ayushs-MacBook-Air:LAB5 iosdeveloper$ vim 4.c
Ayushs-MacBook-Air:LAB5 iosdeveloper$ cc 4.c
Ayushs-MacBook-Air:LAB5 iosdeveloper$ ./a.out
I am Child 5707
I am Parent 5562
Ayushs-MacBook-Air:LAB5 iosdeveloper$ █

```

Demonstrating zombie process being reaped using wait()

```

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

int main(){
pid_t PID=fork();
if (PID>0) {
printf("I am Parent %d I am going to wait for my child\n", getppid());
int child status=wait (NULL);
printf ("I am Parent %d My childs return status is %d\n", getppid(), child status);
}
else if (PID==0){
printf ("I am Child %d\n", getpid());
printf ("My parent is %d\n", getppid () ) ;
exit(0);
}
else{
printf ("ERROR" );
}
return 0;
}

```

```
[Ayushs-MacBook-Air:LAB5 iosdeveloper$ vim 5.c  
[Ayushs-MacBook-Air:LAB5 iosdeveloper$ cc 5.c  
[Ayushs-MacBook-Air:LAB5 iosdeveloper$ ./a.out  
I am Parent 5562 I am going to wait for my child  
I am Child 5728  
My parent is 5727  
I am Parent 5562 My childs return status is 5728  
Ayushs-MacBook-Air:LAB5 iosdeveloper$ █
```