

**Trainee name : Khushi Mordani**  
**Employee id : 150111**  
**Linux Internals – Process Assignment**

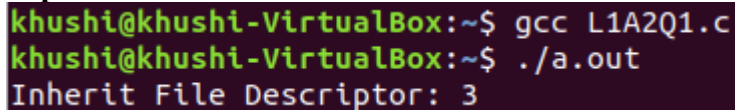
**1.Test whether the process(exec() system call) that replaces old program data , will inherit the fd's or not.**

**Source code:**

```
//Khushi Mordani
//1.Test whether the process(exec() system call) that replaces old program data , will inherit the fd's
or not.
#include<stdio.h>
#include<fcntl.h>
#include<sys/types.h>
#include<sys/stat.h>
#include<unistd.h>
#include<stdlib.h>

int main()
{
    int fd;
    fd=open("data.txt",O_RDONLY,0777);
    printf("Inherit File Descriptor: %d\n",fd);
    return 0;
}
```

**Output:**



```
khushi@khushi-VirtualBox:~$ gcc L1A2Q1.c
khushi@khushi-VirtualBox:~$ ./a.out
Inherit File Descriptor: 3
```

**2.Write a program such that parent process create two child processes,such that each child executes a separate task.**

**Source code:**

```
//Khushi Mordani
//2.Write a program such that parent process create two child processes,such that each child
executes a separate task.
```

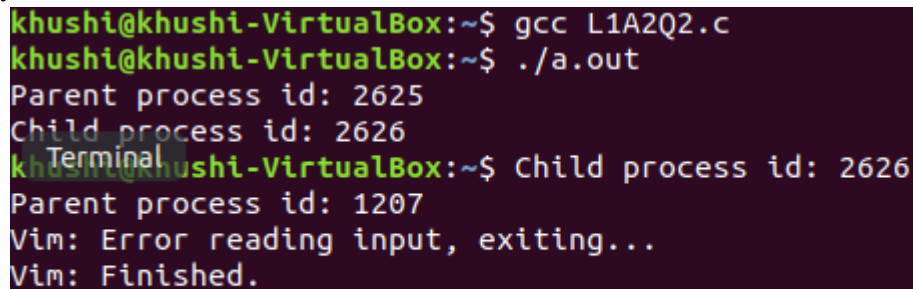
```
#include <stdio.h>
#include<sys/types.h>
#include<unistd.h>
int main()
{
    pid_t q,q_child;
    q = fork();
    if(q==0)
    {
        //wait(NULL);
        printf("Child process id: %d\n",getpid());
        printf("Parent process id: %d\n",getppid());
        execlp("vim", "vim", NULL);
    }
```

```

        q_child=fork();
        if(q_child==0)
        {
            printf("Grandchild process id: %d\n",getpid());
            execlp("ls","-a","-s",NULL);
        }
    }
    else
    {
        //wait(NULL);
        printf("Parent process id: %d\n",getpid());
        printf("Child process id: %d\n",q);
    }
    return 0;
}

```

**Output:**



```

khushi@khushi-VirtualBox:~$ gcc L1A2Q2.c
khushi@khushi-VirtualBox:~$ ./a.out
Parent process id: 2625
Child process id: 2626
khushi@khushi-VirtualBox:~$ Child process id: 2626
Parent process id: 1207
Vim: Error reading input, exiting...
Vim: Finished.

```

**3. A program that replaces old program with new program data and is expected to display the currently running processes in a hierarchical tree format.**

**Source code:**

//Khushi Mordani

//3. A program that replaces old program with new program data and is expected to display the currently running processes in a hierarchical tree format.

```

#include<stdio.h>
#include<unistd.h>
int main()
{
    execlp("pstree","pstree",NULL);
    return 0;
}

```

## Output:

```
khushi@khushi-VirtualBox:~$ gcc L1A2Q3.c
khushi@khushi-VirtualBox:~$ ./a.out
systemd├─ModemManager─2*[{ModemManager}]
│├─NetworkManager─dhclient
││└─2*[{NetworkManager}]
│├─accounts-daemon─2*[{accounts-daemon}]
│├─acpid
│├─avahi-daemon─avahi-daemon
│├─boltd─2*[{boltd}]
│├─colord─2*[{colord}]
│├─cron
│├─cups-browsed─2*[{cups-browsed}]
│├─cupsd─dbus
│├─dbus-daemon
│├─firefox├─Isolated Web Co─17*[{Isolated Web Co}]
││├─2*[{Isolated Web Co─14*[{Isolated Web Co}]]
││├─Privileged Cont─14*[{Privileged Cont}]
││├─RDD Process─2*[{RDD Process}]
││├─Socket Process─4*[{Socket Process}]
││├─3*[{Web Content─12*[{Web Content}]]
││├─WebExtensions─14*[{WebExtensions}]
││└─77*[{firefox}]
│├─fwupd─4*[{fwupd}]
│├─gdm3├─gdm-session-wor├─gdm-x-session├─Xorg─{Xorg}
││├─gnome-session-b├─gnome-shell├─ibus-daemon├─ibus-dconf─3*[{ibus-dconf}]
│││├─ibus-engine-sim─2*[{ibus-engine-sim}]
│││└─2*[{ibus-daemon}]
││├─9*[{gnome-shell}]
││├─gsd-a11y-settin─3*[{gsd-a11y-settin}]
││├─gsd-clipboard─2*[{gsd-clipboard}]
││├─gsd-color─3*[{gsd-color}]
││├─gsd-datetime─2*[{gsd-datetime}]
││├─gsd-housekeepin─2*[{gsd-housekeepin}]
││├─gsd-keyboard─3*[{gsd-keyboard}]
││├─gsd-media-keys─3*[{gsd-media-keys}]
││├─gsd-mouse─2*[{gsd-mouse}]
││├─gsd-power─3*[{gsd-power}]
││├─gsd-print-notif─2*[{gsd-print-notif}]
││├─gsd-rfkill─2*[{gsd-rfkill}]
││├─gsd-screensaver─2*[{gsd-screensaver}]
││├─gsd-sharing─3*[{gsd-sharing}]
││├─gsd-smartcard─4*[{gsd-smartcard}]
││└─gsd-sound─3*[{gsd-sound}]
```

4.A processs using execl() system call should replace a new command line program.

## Source code:

//Khushi Mordani

//4.A processs using execl() system call should replace a new command line program.

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

int main(int argc, char* argv[])
{
    // int ret = execl("/usr/bin/vim", "vim","info1.txt", 0);
    int ret = execl(argv[1], argv[2],argv[3], 0);
    if(ret == -1)
    {
        printf("execl returned error %d\n", ret);
    }
    return(0);
}
```

## Output:

```
khushi@khushi-VirtualBox:~$ gcc L1A2Q4.c
L1A2Q4.c: In function 'main':
L1A2Q4.c:11:2: warning: missing sentinel in function call [-Wformat=]
   int ret = execl(argv[1], argv[2], argv[3], 0);
   ^~~~
khushi@khushi-VirtualBox:~$ ./a.out /usr/bin/pstree pstree
systemd--ModemManager--2*[{ModemManager}]
      |--NetworkManager--dhclient
      |                   2*[{NetworkManager}]
      |--accounts-daemon--2*[{accounts-daemon}]
      |--acpid
      |--avahi-daemon--avahi-daemon
      |--boltd--2*[{boltd}]
      |--colord--2*[{colord}]
      |--cron
      |--cups-browsed--2*[{cups-browsed}]
      |--cupsd--dbus
      |--dbus-daemon
      |--firefox--Isolated Web Co--17*[{Isolated Web Co}]
      |           2*[{Isolated Web Co}]--14*[{Isolated Web Co}]
      |           Isolated Web Co--18*[{Isolated Web Co}]
      |           Privileged Cont--14*[{Privileged Cont}]
      |           RDD Process--2*[{RDD Process}]
      |           Socket Process--4*[{Socket Process}]
      |           WebExtensions--14*[{WebExtensions}]
      |           103*[{firefox}]
      |--fwupd--4*[{fwupd}]
      |--gdm3--gdm-session-wor--gdm-x-session--Xorg--{Xorg}
      |                                     |--gnome-session-b--gnome-shell+
      |                                     |--gsd-a11y-s+
      |                                     |--gsd-clipbo+
      |                                     |--gsd-color-+++
      |                                     |--gsd-dateti+
      |                                     |--gsd-housek+
      |                                     |--gsd-keyboa+
      |                                     |--gsd-media-+
      |                                     |--gsd-mouse-+++
      |                                     |--gsd-power-+++
      |                                     |--gsd-print-+
```

**5. Write a program parent process wait until, while child process open a file and read file data into empty buffer.**

### Source code:

//Khushi Mordani

//5. Write a program parent process wait until, while child process open a file and read file data into empty buffer.

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

```
int main()
{
    int pid;
    char read_buf[100];
    pid=fork();
    if(pid==0)
    {
        //child process
        printf("Child process is running \n");
        int fd;
        fd = open("readfile.c",O_RDONLY);
        read(fd,read_buf,100);
        printf("%s",read_buf);
        //it will read data from the readfile and it will print on display
    }
}
```

```

        printf("Child process stopped...!\n\n");
    }
    else
    {
        wait(0);//parent will wait till child execute
        //sleep(5);
        printf("Parent running\n");
        printf("Parent Process Stopped...!\n");
    }
    return 0;
}

```

### Output:

```

khushi@khushi-VirtualBox:~$ gcc L1A2Q5.c
L1A2Q5.c: In function 'main':
L1A2Q5.c:26:3: warning: implicit declaration of function 'wait'; did you mean 'main'? [-Wimplicit-function-declaration]
    wait(0);//parent will wait till child execute
    ^~~~~
    main
khushi@khushi-VirtualBox:~$ ./a.out
Child process is running
Child process stopped...!

Parent running
Parent Process Stopped...!

```

**5. Write a program, where functions of the program are called in the reverse order of their function calls from main().**

### Source code:

//Khushi Mordani

//5. Write a program, where functions of the program are called in the reverse order of their function calls from main().

```

#include<stdio.h>
#include<stdlib.h>

void callBack1()
{
    printf("callback 1\n");
}
void callBack2()
{
    printf("callback 2\n");
}
void callBack3()
{
    printf("callback 3\n");
}

int main()
{
    printf("registering callback1\n");
    atexit(callBack1);
    printf("registering callback2\n");
    atexit(callBack2);
    printf("registering callback3\n");
    atexit(callBack3);
}

```

```

    printf("exit() main exiting now...\n");
    exit(0); //_exit() is not calling the functions which are previously registered using atexit
function
}

```

#### Output:

```

khushi@khushi-VirtualBox:~$ gcc L1A2Q5-2.c
khushi@khushi-VirtualBox:~$ ./a.out
registering callback1
registering callback2
registering callback3
exit() main exiting now...
callback 3
callback 2
callback 1

```

**6. Write a program child executes(exec()) a new program , while parent waits for child task to get complete.**

#### Source code:

//Khushi Mordani

//6. Write a program child executes(exec()) a new program , while parent waits for child task to get complete.

```

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

```

```

int main()
{
    int pid;
    pid=fork();
    if(pid==0)
    {
        //child process
        printf("Child process is running \n");
        int ret = execl("/home/ompatel/Linux_Internals/10-03-22/Asignment/6","",NULL);
        //5 file will execute
    }
    else
    {
        wait(2);//parent will wait till child execute
        //sleep(5);
        printf("Parent running\n");
        printf("Parent Process Stopped...\n");
    }
    return 0;
}

```

### Output:

```
khushi@khushi-VirtualBox:~$ gcc L1A2Q6.c
L1A2Q6.c: In function 'main':
L1A2Q6.c:20:3: warning: implicit declaration of function 'wait'; did you mean 'main'? [-Wimplicit-function-declaration]
    wait(2); //parent will wait till child execute
    ^~~~~
    main
khushi@khushi-VirtualBox:~$ ./a.out
Child process is running
Parent running
Parent Process Stopped...!
```