

→ JOSHI MAYURKUMAR A

LINUX INTERNAL ASSIGNMENT - Linux Internals: Process Assignment

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

CODE—

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

int main(int argc, char *argv[])
{
    printf("PID of first = %d\n", getpid());
    char *args[] = {"MAYUR", "JOSHI", "EMBEDED",
NULL};
    execv("./L1b",args);
    printf("Back to first");

    return 0;
}
```

OUTPUT—

```
mayur@mayur-VirtualBox:~$ gcc lin1.c
mayur@mayur-VirtualBox:~$ ./a.out
PID of first = 2719
Back to firstmayur@mayur-VirtualBox:~$ gedit lin1.c
```

Q1.(b):

CODE—

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

int main(int argc, char *argv[])
{
    printf("here we have\n");
    printf("PID of first= %d\n" , getpid());

    return 0;
}
```

OUTPUT—

```
mayur@mayur-VirtualBox:~$ gcc lin1b.c
mayur@mayur-VirtualBox:~$ ./a.out
here we have
PID of first= 2749
mayur@mayur-VirtualBox:~$
```

Q2:Write a program such that parent process create two child processes,such that each child executes a separate task.

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

```
int main()
{
    int pid1, pid2;

    printf("Curent Process id: %d\n",getpid());
    // printf("parent process pid: %d\n",getppid());

    pid1 = fork();
    printf("Pid1: %d\n",pid1);

    if(pid1 == 0)
    {
        printf("Child 1 process id is: %d\n",getpid());
        printf("Child 1 Parent process id is: %d\n",getppid());

        for(int i=0; i<5 ; i++)
        {
            printf("first child process : %d\n",i);
        }

        // while(1);
    }
}
```

```
else
{
    pid2 = fork();

    if(pid2 == 0)
    {
        printf("Child 2 process id is: %d\n",getpid());
        printf("Child 2 Parent process id is: %d\n",getppid());

        int a=10, b=20, result=0;

        result=a+b;
        printf("The result is: %d\n",result);

        //while(1);
    }
    else
    {
        printf("Parent's pid is: %d\n",getpid());
        printf("Parent's ppid is: %d\n",getppid());
    }
}

printf("End of program\n");

return 0;

}
```

OUTPUT—

```
mayur@mayur-VirtualBox:~$ gcc lin2.c
mayur@mayur-VirtualBox:~$ ./a.out
Curent Process id: 2782
Pid1: 2783
Parent's pid is: 2782
Parent's ppid is: 2083
End of program
mayur@mayur-VirtualBox:~$ Child 2 process id is: 2784
Child 2 Parent process id is: 1341
The result is: 30
End of program
Pid1: 0
Child 1 process id is: 2783
Child 1 Parent process id is: 1341
first child process : 0
first child process : 1
first child process : 2
first child process : 3
first child process : 4
End of program
```

Q3: A program that replaces old program with new program data and is expected to display the currently running processes in a hierarchical tree format.

CODE—

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

int main()
{
    printf("In the current process, pid is: %d\n",getpid());

    printf("Replacing old program with new data by using execl()
system call\n");
    execl("/usr/bin/pstree", "pstree", 0);

    printf("Exiting main program\n");

    return 0;
}
```

OUTPUT—

```
mayur@mayur-VirtualBox:~$ ./a.out
In the current process, pid is: 2820
Replacing old program with new data by using execl() system call
systemd--ModemManager--2*[{ModemManager}]
--NetworkManager--dhclient
--2*[{NetworkManager}]
--3*[VBoxClient--VBoxClient--2*[{VBoxClient}]]
--VBoxClient--VBoxClient--3*[{VBoxClient}]
--VBoxService--8*[{VBoxService}]
--accounts-daemon--2*[{accounts-daemon}]
--acpid
--apache2--2*[apache2--26*[{apache2}]]
--avahi-daemon--avahi-daemon
--boltd--2*[{boltd}]
--colord--2*[{colord}]
--cron
--cups-browsed--2*[{cups-browsed}]
--cupsd--dbus
--dbus-daemon
--fwupd--4*[{fwupd}]
--gdm3--gdm-session-wor--gdm-wayland-ses--gnome+
--2*[{gdm-session-wor}]
--2*[{gdm-session-wor}]--gdm-x-session--Xorg--+
--gnome-s+
--2*[{gdm-session-wor}]
--2*[{gdm3}]
--gnome-kevrina-d--3*[{gnome-kevrina-d}]
```

```

└─2*[{gdm3}]
gnome-keyring-d—3*[{gnome-keyring-d}]
gsd-printer—2*[{gsd-printer}]
2*[{ibus-x11}—2*[{ibus-x11}]]
2*[{kerneloops}]
networkd-dispat—{networkd-dispat}
packagekitd—2*[{packagekitd}]
polkitd—2*[{polkitd}]
pulseaudio—3*[{pulseaudio}]
rsyslogd—3*[{rsyslogd}]
rtkit-daemon—2*[{rtkit-daemon}]
snapd—9*[{snapd}]
systemd—(sd-pam)
└─at-spi-bus-laun—dbus-daemon
└─3*[{at-spi-bus-laun}]+
└─at-spi2-registr—2*[{at-spi2-registr}]+
└─dbus-daemon
└─ibus-portal—2*[{ibus-portal}]
└─pulseaudio—3*[{pulseaudio}]
└─xdg-permission-—2*[{xdg-permission-}]+
systemd—(sd-pam)
└─at-spi-bus-laun—dbus-daemon
└─3*[{at-spi-bus-laun}]+
└─at-spi2-registr—2*[{at-spi2-registr}]+
└─dbus-daemon
└─dconf-service—2*[{dconf-service}]
└─evolution-addre—evolution-addre—5*+
└─4*[{evolution-addre}]+
└─evolution-calen—evolution-calen—8*+

```

```

└─evolution-sourc—3*[{evolution-sourc}]+
└─gnome-shell-cal—5*[{gnome-shell-cal}]+
└─gnome-terminal-└─bash—pstree
└─3*[{gnome-terminal-}]+
└─goa-daemon—3*[{goa-daemon}]
└─goa-identity-se—3*[{goa-identity-se}]+
└─gvfs-afc-volume—3*[{gvfs-afc-volume}]+
└─gvfs-goa-volume—2*[{gvfs-goa-volume}]+
└─gvfs-gphoto2-vo—2*[{gvfs-gphoto2-vo}]+
└─gvfs-mtp-volume—2*[{gvfs-mtp-volume}]+
└─gvfs-udisks2-vo—2*[{gvfs-udisks2-vo}]+
└─gvfsd└─gvfsd-trash—2*[{gvfsd-trash}]+
└─2*[{gvfsd}]
└─gvfsd-fuse—5*[{gvfsd-fuse}]
└─gvfsd-metadata—2*[{gvfsd-metadata}]
└─ibus-portal—2*[{ibus-portal}]
└─nautilus—4*[{nautilus}]
└─xdg-permission-—2*[{xdg-permission-}]+
systemd-journal
systemd-logind
systemd-resolve
systemd-udev
udisksd—5*[{udisksd}]
unattended-upgr—{unattended-upgr}
upowerd—2*[{upowerd}]
whoopsie—2*[{whoopsie}]
wpa_supplicant

```

mayur@mayur-VirtualBox:~\$

Q4: A processs using execl() system call should replace a new command line program.

CODE—

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

int main()
{
    printf("This is the program 4\n");

    execl("/bin/ls", "ls", "-lh", "-a" , 0);

    return 0;
}
```

OUTPUT—

```
mayur@mayur-VirtualBox:~$ ./a.out
This is the program 4
total 1.1M
-rw-rw-r-- 1 mayur mayur 0 Feb 28 10:53 '='
drwxr-xr-x 23 mayur mayur 12K Mar 18 11:24 .
drwxr-xr-x 3 root root 4.0K Jan 24 19:29 ..
-rw-rw-r-- 1 mayur mayur 0 Feb 28 10:53 '=10'
-rw-rw-r-- 1 mayur mayur 0 Feb 28 11:15 '=80'
-rw-rw-r-- 1 mayur mayur 0 Feb 28 10:52 9
-rw-rw-r-- 1 mayur mayur 26 Mar 18 10:20 abc.txt
-rw-rw-r-- 1 mayur mayur 715 Mar 10 12:46 abstect.c
-rw-rw-r-- 1 mayur mayur 715 Mar 10 12:47 abstect.cpp
-rw-rw-r-- 1 mayur mayur 239 Mar 16 09:34 alloca.c
-rw-rw-r-- 1 mayur mayur 260 Mar 16 09:39 alloc.c
-rwxrwxr-x 1 mayur mayur 8.2K Mar 18 11:24 a.out
-rw-rw-r-- 1 mayur mayur 1.2K Mar 5 20:51 array.ds
-rw-rw-r-- 1 mayur mayur 544 Mar 15 14:54 assi1
-rw-rw-r-- 1 mayur mayur 552 Mar 15 14:56 assi1.c
-rw-rw-r-- 1 mayur mayur 318 Mar 18 10:43 assi1.cpp
-rw-rw-r-- 1 mayur mayur 616 Mar 15 14:58 assi2.c
-rw-rw-r-- 1 mayur mayur 353 Mar 15 14:59 assi3.c
-rw-rw-r-- 1 mayur mayur 566 Mar 15 15:01 assi4.c
-rw-rw-r-- 1 mayur mayur 514 Mar 15 15:05 assi5.c
-rw-rw-r-- 1 mayur mayur 640 Mar 8 08:04 assig.cpp
-rw-rw-r-- 1 mayur mayur 501 Mar 10 08:51 atexit.c
-rw-r--r-- 1 mayur mayur 20K Mar 17 13:12 .bash_history
-rw-r--r-- 1 mayur mayur 220 Jan 24 19:29 .bash_logout
-rw-r--r-- 1 mayur mayur 3.7K Jan 24 19:29 .bashrc
-rw-rw-r-- 1 mayur mayur 1.1K Mar 5 20:52 binary.ds
```



```

-rw-rw-r-- 1 mayur mayur 209 Mar 17 11:56 tem.cpp
drwxr-xr-x 2 mayur mayur 4.0K Jan 24 19:39 Templates
-rw-rw-r-- 1 mayur mayur 0 Feb 28 12:30 test
drwxrwxr-x 3 mayur mayur 4.0K Feb 6 21:15 test1
drwxrwxr-x 2 mayur mayur 4.0K Feb 6 21:43 test3
-rw-rw-r-- 1 mayur mayur 269 Mar 10 11:56 thispointer.cpp
-rw-rw-r-- 1 mayur mayur 721 Mar 10 10:29 thread1.c
drwx----- 6 mayur mayur 4.0K Jan 27 19:26 .thunderbird
-rw-rw-r-- 1 mayur mayur 2.1K Feb 18 22:43 timeanddate.c++
-rw-rw-r-- 1 mayur mayur 3.4K Feb 16 22:16 tree
-rw-rw-r-- 1 mayur mayur 3.4K Feb 16 22:16 tree.c
-rw-rw-r-- 1 mayur mayur 4.1K Mar 5 20:53 tree.ds
-rw-rw-r-- 1 mayur mayur 437 Feb 16 12:13 twoobj
-rw-rw-r-- 1 mayur mayur 436 Feb 16 12:19 twoobj.c++
-rwxr--r-- 1 mayur mayur 158 Feb 28 12:34 uid.sh
-rw-r----- 1 mayur mayur 5 Mar 18 10:00 .vboxclient-clipboard.pid
-rw-r----- 1 mayur mayur 5 Mar 18 10:00 .vboxclient-display-svg-x11.pid
-rw-r----- 1 mayur mayur 5 Mar 18 10:00 .vboxclient-draganddrop.pid
-rw-r----- 1 mayur mayur 5 Mar 18 10:00 .vboxclient-seamless.pid
drwxr-xr-x 2 mayur mayur 4.0K Jan 24 19:39 Videos
-rw-rw-r-- 1 mayur mayur 165 Mar 9 10:44 vineditor.c
-rw----- 1 mayur mayur 1.5K Mar 9 10:10 .viminfo
-rw-rw-r-- 1 mayur mayur 358 Mar 10 12:32 virtual.c
-rw-rw-r-- 1 mayur mayur 318 Mar 8 13:10 virtual.cpp
-rw-rw-r-- 1 mayur mayur 532 Mar 10 09:14 waitpid.c
-rw-rw-r-- 1 mayur mayur 412 Mar 10 09:17 waitpidc.c
-rw-rw-r-- 1 mayur mayur 242 Mar 8 09:38 write.c
mayur@mayur-VirtualBox:~$

```

Q5:Write a program parent process wait untill ,while child process open a file and read file data into empty buffer.

CODE—

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

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

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

```
int main()
```

```
{
```

```
    int fd, pid;
```

```
    char buff[300];
```

```
    printf("Current process pid is: %d\n",getpid());
```

```
    pid = fork();
```

```
    printf("parent process pid is: %d, PID value is: %d\n",getpid(), pid);
```

```
    if(pid == 0)
```

```
    {
```

```
        printf("Child process pid: %d\n",getpid());
```

```

        fd = open("child_file.txt", O_CREAT|O_RDWR, 0666);

        printf("Value of fd: %d\n",fd);


        printf("Writing data to empty file\n");
        read(fd, buff, sizeof(buff));


        printf("\nfile content is: \n%s\n",buff);
    }
    else
    {

        printf("This is parent process, pid is: %d\n",getpid());
        printf("Parent's ppid is: %d\n",getppid());


        sleep(4);
    }


    return 0;
}

```

OUTPUT—

```

mayur@mayur-VirtualBox:~$ gcc lin5.c
mayur@mayur-VirtualBox:~$ ./a.out
Current process pid is: 2921
parent process pid is: 2921, PID value is: 2922
This is parent process, pid is: 2921
Parent's ppid is: 2083
parent process pid is: 2922, PID value is: 0
Child process pid: 2922
Value of fd: 3
Writing data to empty file

file content is:
        ~
mayur@mayur-VirtualBox:~$

```

Q6:Write a program, where functions of the program are called in the reverse order of their function calls from main().

CODE—

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

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

```
int x, y;
```

```
void looping()
```

```
{
```

```
    for(int i=0; i<5; i++)
```

```
        printf("In loop value: %d\n",i*2);
```

```
}
```

```
void addition()
```

```
{
```

```
    int sum=0;
```

```
    sum = x+y;
```

```
    printf("Result of addition is: %d\n",sum);
```

```
}
```

```
void getValue()
```

```
{
```

```
    printf("Enter two Values:\n");
```

```
    scanf("%d",&x);
```

```
    scanf("%d",&y);
```

```
}
```

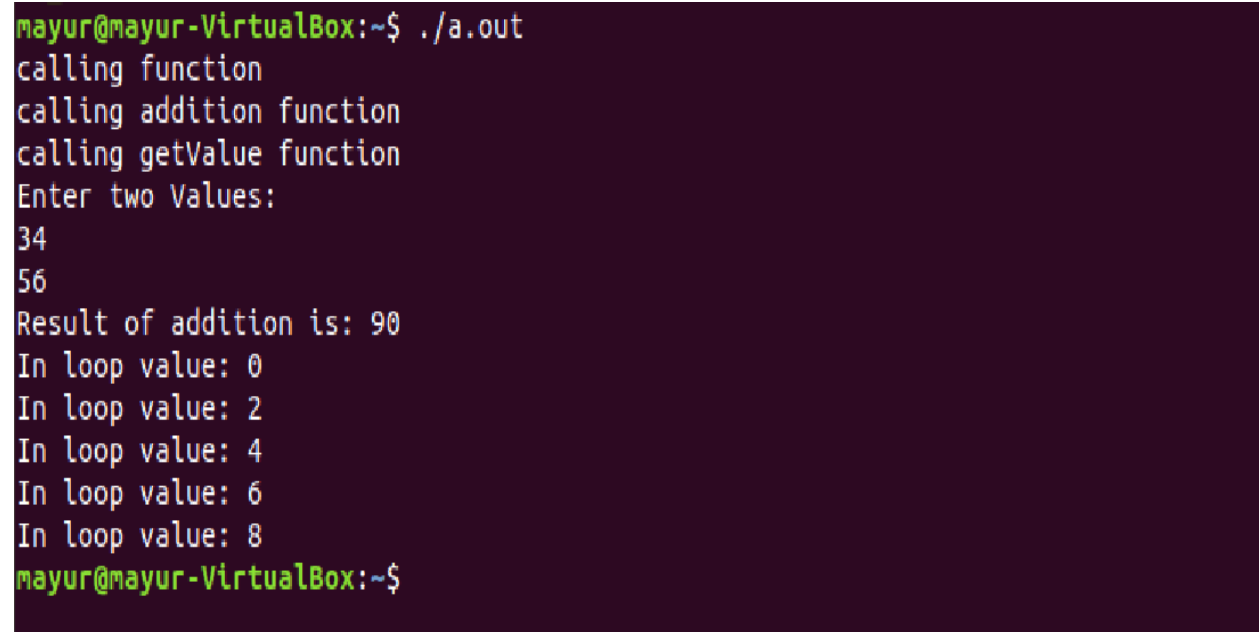
```
int main()
{
    printf("calling function\n");
    atexit(looping);

    printf("calling addition function\n");
    atexit(addition);

    printf("calling getValue function\n");
    atexit(getValue);

    return 0;
}
```

OUTPUT—

A terminal window with a dark purple background and light green text. The prompt is 'mayur@mayur-VirtualBox:~\$'. The user enters './a.out'. The program outputs 'calling function', 'calling addition function', and 'calling getValue function'. It then prompts 'Enter two Values:' and the user enters '34' and '56'. The program outputs 'Result of addition is: 90' and then enters a loop printing 'In loop value: 0', 'In loop value: 2', 'In loop value: 4', 'In loop value: 6', and 'In loop value: 8'. The prompt returns to 'mayur@mayur-VirtualBox:~\$'.

```
mayur@mayur-VirtualBox:~$ ./a.out
calling function
calling addition function
calling getValue function
Enter two Values:
34
56
Result of addition is: 90
In loop value: 0
In loop value: 2
In loop value: 4
In loop value: 6
In loop value: 8
mayur@mayur-VirtualBox:~$
```

Q7: Write a program child executes(exec()) a new program , while parent waits for child task to get complete.

CODE—

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

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

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

```
int main()
```

```
{
```

```
    int fd, pid;
```

```
    printf("Current process pid is: %d\n",getpid());
```

```
    pid = fork();
```

```
    printf("parent process pid is: %d, value of pid is: %d\n",getpid(), pid);
```

```
    if(pid == 0)
```

```
    {
```

```
        printf("Child process pid: %d\n",getpid());
```

```
        execl("/home/ubuntu/ubuntu/einfochips/Final_project/linux_internals_assignment/process_assignment/7_table7", "./7_table7", 0);
```

```
    }
```

```
    else
```

```
    {
```

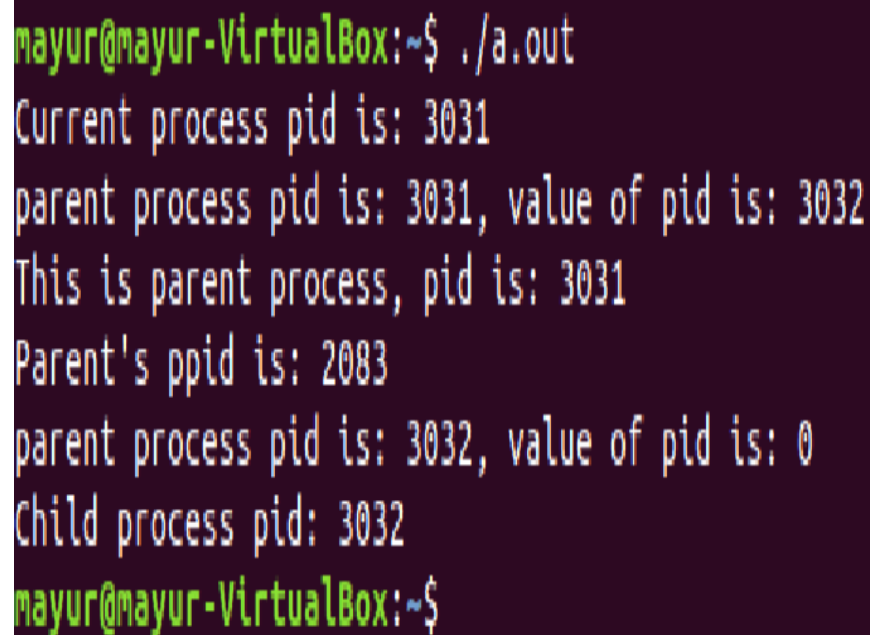
```
        printf("This is parent process, pid is: %d\n",getpid());
```

```
        printf("Parent's ppid is: %d\n",getppid());
```

```
        sleep(5);
```

```
    }  
  
    return 0;  
}
```

OUTPUT—

A screenshot of a terminal window with a dark purple background. The prompt is 'mayur@mayur-VirtualBox:~\$'. The user has entered './a.out'. The output of the program is displayed in a monospaced font: 'Current process pid is: 3031', 'parent process pid is: 3031, value of pid is: 3032', 'This is parent process, pid is: 3031', 'Parent's ppid is: 2083', 'parent process pid is: 3032, value of pid is: 0', and 'Child process pid: 3032'. The prompt 'mayur@mayur-VirtualBox:~\$' appears again at the bottom.

```
mayur@mayur-VirtualBox:~$ ./a.out  
Current process pid is: 3031  
parent process pid is: 3031, value of pid is: 3032  
This is parent process, pid is: 3031  
Parent's ppid is: 2083  
parent process pid is: 3032, value of pid is: 0  
Child process pid: 3032  
mayur@mayur-VirtualBox:~$
```