



EXECCL >  execExam.cpp >  main(int, char \* [])

```
1  #include <iostream>
2  #include <unistd.h>
3  #include <sys/types.h>
4  #include <stdlib.h>
5  #include <sys/wait.h>
6  #include <errno.h>
7  #include <stdio.h>
8  using namespace std;
9
10
11  // Muneel Haider
12  // 21i-0640
13
14  int main(int argc, char* argv[]){
15
16      for(int i = 0; i < argc; i++){
17
18          printf("Argument %d : %s \n", i, argv[i]);
19      }
20  }
```

PROBLEMS



2

DEBUG CONSOLE

TERMINAL

Argument 0 : /media/muneelhaider/68909D22909CF7B0/Softwares/Visual  
[1] + Done

"/usr/bin/gdb" --interpreter=mi  
muneelhaider@MuneelHaiderLinux:~/Desktop/OS/LAB\_4\$

EXECL >  execl\_1.cpp >  main()

```
8   using namespace std;
9
10
11  // Muneel Haider
12  // 21i-0640
13
14
15  // Example EXECL
16  int main(){
17
18      pid_t childpid = fork();
19
20      if(childpid == 0){
21
22          printf("I am child process with pid = %d \n", getpid());
23          printf("The next statement is execl and ls will \n");
24
25          execl("/bin/ls" ,"ls", "-l", "/usr", NULL);
26
27          printf("Execl Failed \n");
28
29      }
30      else if(childpid > 0){
31
32          wait(NULL);
33          printf("\nI am parent process with pid = %d and finished waiting \n", getpid());
34      }
35
36      return 0;
37  }
```

PROBLEMS 2 DEBUG CONSOLE TERMINAL



```
I am child process with pid = 69798
The next statement is execl and ls will
bin
games
include
lib
lib32
lib64
libexec
libx32
local
sbin
share
src
```

```
I am parent process with pid = 69792 and finished waiting
```

```
[1] + Done
```

```
"/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<"/tmp/Microsoft-MIEng
```

```
muneelhaider@MuneelHaiderLinux:~/Desktop/OS/LAB_4$
```

EXECL >  execl\_2.cpp >  main()

```
5  #include <sys/wait.h>
6  #include <errno.h>
7  #include <stdio.h>
8  using namespace std;
9
10
11  // Muneel Haider
12  // 21i-0640
13
14  int main(){
15
16      pid_t childpid = fork();
17
18      if (childpid == 0) {
19
20          printf("I am child process with pid = %d \n", getpid());
21          printf("The next statement is execl and ls will \n");
22
23          execl("/home/muneelhaider/Desktop/OS/LAB_4/EXECL/execExam", "ls", "-l", "/usr", NULL);
24          printf("Execl Failed.");
25      }
26
27      else if(childpid = 0){
28
29          wait(NULL);
30          printf("\nI am parent process with pid = %d and finished waiting\n", getpid());
31      }
32
33      return 0;
34  }
```



PROBLEMS 2 DEBUG CONSOLE TERMINAL

```
I am child process with pid = 70373
The next statement is execl and ls will
Argument 0 : ls
Argument 1 : -l
Argument 2 : /usr
```

```
[1] + Done "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<"/tmp/Microsoft-MIEngine-
```

```
muneelhaider@MuneelHaiderLinux:~/Desktop/OS/LAB_4$
```

```

EXECLP >  execlp.cpp >  main()
8   using namespace std;
9
10
11   // Muneel Haider
12   // 21i-0640
13
14
15   // Example EXECL
16   int main(){
17
18       pid_t childpid = fork();
19
20       if(childpid == 0){
21
22           printf("I am child process with pid = %d \n", getpid());
23           printf("The next statement is execl and ls will \n");
24
25           execlp("ls", "ls", "-l", "/usr", NULL);
26
27           printf("Execlp Failed");
28       }
29
30       else if(childpid > 0){
31
32           wait(NULL);
33           printf("\nI am parent process with pid %d and finished waiting\n", getpid());
34       }
35
36       return 0;
37   }

```

PROBLEMS 2 DEBUG CONSOLE TERMINAL

```



I am child process with pid = 70529
The next statement is execl and ls will
bin
games
include
lib
lib32
lib64
libexec
libx32
local
sbin
share
src

```

```

I am parent process with pid 70523 and finished waiting
[1] + Done                               "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<"/tmp/Microso
muneelhaider@MuneelHaiderLinux:~/Desktop/OS/LAB_4$

```

EXECV >  execv.cpp >  main()

```
8 using namespace std;
9
10
11 // Muneel Haider
12 // 21i-0640
13
14 // Example EXECV
15 int main(){
16
17     pid_t childpid = fork();
18
19     if(childpid == 0){
20
21         printf("I am child process with pid = %d \n", getpid());
22         printf("The next statement is execv and ls will \n");
23
24         char* argv[] = { "ls", "-l", "/usr", NULL};
25
26         execv("/bin/ls", argv);
27         printf("Execv Failed");
28     }
29
30     else if(childpid > 0){
31
32         wait(NULL);
33         printf("\nI am parent process with pid = %d and finished waiting \n", getpid());
34     }
35
36     return 0;
37 }
```

PROBLEMS 5 DEBUG CONSOLE TERMINAL

I am child process with pid = 70658  
The next statement is execv and ls will  
bin  
games  
include  
lib  
lib32  
lib64  
libexec  
libx32  
local  
sbin  
share  
src

I am parent process with pid = 70651 and finished waiting  
[1] + Done "/usr/bin/gdb" --interpreter=mi --tty=\${DbgTerm} 0<"/tmp/Microsoft-M  
muneelhaider@MuneelHaiderLinux:~/Desktop/OS/LAB\_4\$

```
10
11 // Muneel Haider
12 // 21i-0640
13
14
15 // Example EXECVP
16 int main(){
17
18     pid_t childpid = fork();
19
20     if(childpid == 0){
21
22         printf("I am child process with pid = %d \n", getpid());
23         printf("The next statement is execel and ls will \n");
24
25         char* argv[] = { "ls", "-l", "/usr", NULL };
26
27         execvp("ls", argv);
28
29         printf("Execvp Failed. \n");
30     }
31
32     else if(childpid > 0){
33
34         wait(NULL);
35         printf("\nI am parent process with pid = %d and finished waiting \n", getpid());
36     }
37
38     return 0;
39 }
```

PROBLEMS 5 DEBUG CONSOLE TERMINAL

```
I am child process with pid = 70773
The next statement is execel and ls will
bin
games
include
lib
lib32
lib64
libexec
libx32
local
sbin
share
src
```

```
I am parent process with pid = 70767 and finished waiting
```

```
[1] + Done          "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<"/tmp/Microsoft-MI
```

```
muneelhaider@MuneelHaiderLinux:~/Desktop/OS/LAB_4$
```