SYSCALL Muhammed Nifal

Assignment: 3 CB.EN.P2CYS23017

1.GETPID

```
[08/23/23]seed@VM:~$ gcc -g syscall.c
gcc: error: syscall.c: No such file or directory
gcc: fatal error: no input files
compilation terminated.
[08/23/23]seed@VM:~$ cd desktop
bash: cd: desktop: No such file or directory
[08/23/23]seed@VM:~$ cd Desktop
[08/23/23]seed@VM:~/Desktop$ gcc -g syscall.c
[08/23/23]seed@VM:~/Desktop$ ./a.out
syscall(SYS_getpid)=25159
getpid()=25159
[08/23/23]seed@VM:~/Desktop$
```

```c
#include <syscall.h>
#include <unistd.h>
#include <stdio.h>
#include <sys/types.h>
int main(void)
{
long ID1, ID2;
/* direct system call - SYS_getpid (func
no. is 20) */
ID1 = syscall(SYS_getpid);
printf ("syscall(SYS_getpid)=%ld\n",
ID1);
/* "libc" wrapped system call */
/* SYS_getpid (Func No. is 20) */
ID2 = getpid();
printf ("getpid()=%ld\n", ID2);
return(0);
}
```

2. FORK.C

```c
#include <syscall.h>
#include <unistd.h>
#include <stdio.h>
#include <sys/types.h>
main()
{
int return_value;
printf("Forking process\n");
return_value = fork();
printf("The process id is %d and return value is %d\n", getpid(), return_value);
printf("This line is not printed\n")
}
```

```
[08/23/23]seed@VM:~/Desktop$ gedit fork.c
[08/23/23]seed@VM:~/Desktop$ gcc -g fork.c -o fork
fork.c:5:1: warning: return type defaults to ‘int’ [-Wimplicit-int]
    5 | main()
      | ^~~~
[08/23/23]seed@VM:~/Desktop$ ./fork
Forking process
The process id is 25393 and return value is 25394
This line is not printed
The process id is 25394 and return value is 0
This line is not printed
[08/23/23]seed@VM:~/Desktop$
```

3. FORK with If Else.

```c
#include <syscall.h>
#include <unistd.h>
#include <stdio.h>
#include <sys/types.h>
int main()
{
int return_val;
printf("Forking Process\n");
return_val = fork();
if (return_val==0)
{
printf(" Child PRocess: The process ID is %d and return Value is %d \n", getpid(),return_val);
}
else {
printf(" Parent Process: The process ID is %d and return value is %d \n", getpid(),return_val);
}
return 0;
}

```

```
08/23/23]seed@VM:~/Desktop$ gcc -g forkelse.c
08/23/23]seed@VM:~/Desktop$ ./a.out
orking Process
Parent Process: The process ID is 25691 and return value is 25692
Child PRocess: The process ID is 25692 and return Value is 0
08/23/23]seed@VM:~/Desktop$
```

4. EXEC.C

```c
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
int main(int argc, char *argv[])
{
printf("PID = %d\n", getpid());
char *args[] = {"Hello", "C", "Programming", NULL};
execv("./hello", args);
printf("Back to exec.c - This line will not be executed");
return 0;
}
```

```
[08/23/23]seed@VM:~/Desktop$ gedit forkelse.c
[08/23/23]seed@VM:~/Desktop$ gedit exec.c
[08/23/23]seed@VM:~/Desktop$ gcc -g forkelse.c
[08/23/23]seed@VM:~/Desktop$ gcc -g exec.c
[08/23/23]seed@VM:~/Desktop$ ./a.out
PID = 25811
Back to exec.c - This line will not be executed[08/23/23]seed@VM:~/Desktop$
```

Hello.C

```c
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
int main(int argc, char *argv[])
{
printf("We are in Hello.c\n");
printf("PID of hello.c = %d\n", getpid());
return 0;
}
```

```
[08/23/23]seed@VM:~$ cd Desktop
[08/23/23]seed@VM:~/Desktop$ gedit hello.c
[08/23/23]seed@VM:~/Desktop$ gcc -g hello.c -o hello
[08/23/23]seed@VM:~/Desktop$ gcc -g exec.c
[08/23/23]seed@VM:~/Desktop$ ./a.out
PID = 25911
We are in Hello.c
PID of hello.c = 25911
[08/23/23]seed@VM:~/Desktop$
```