Lab-9

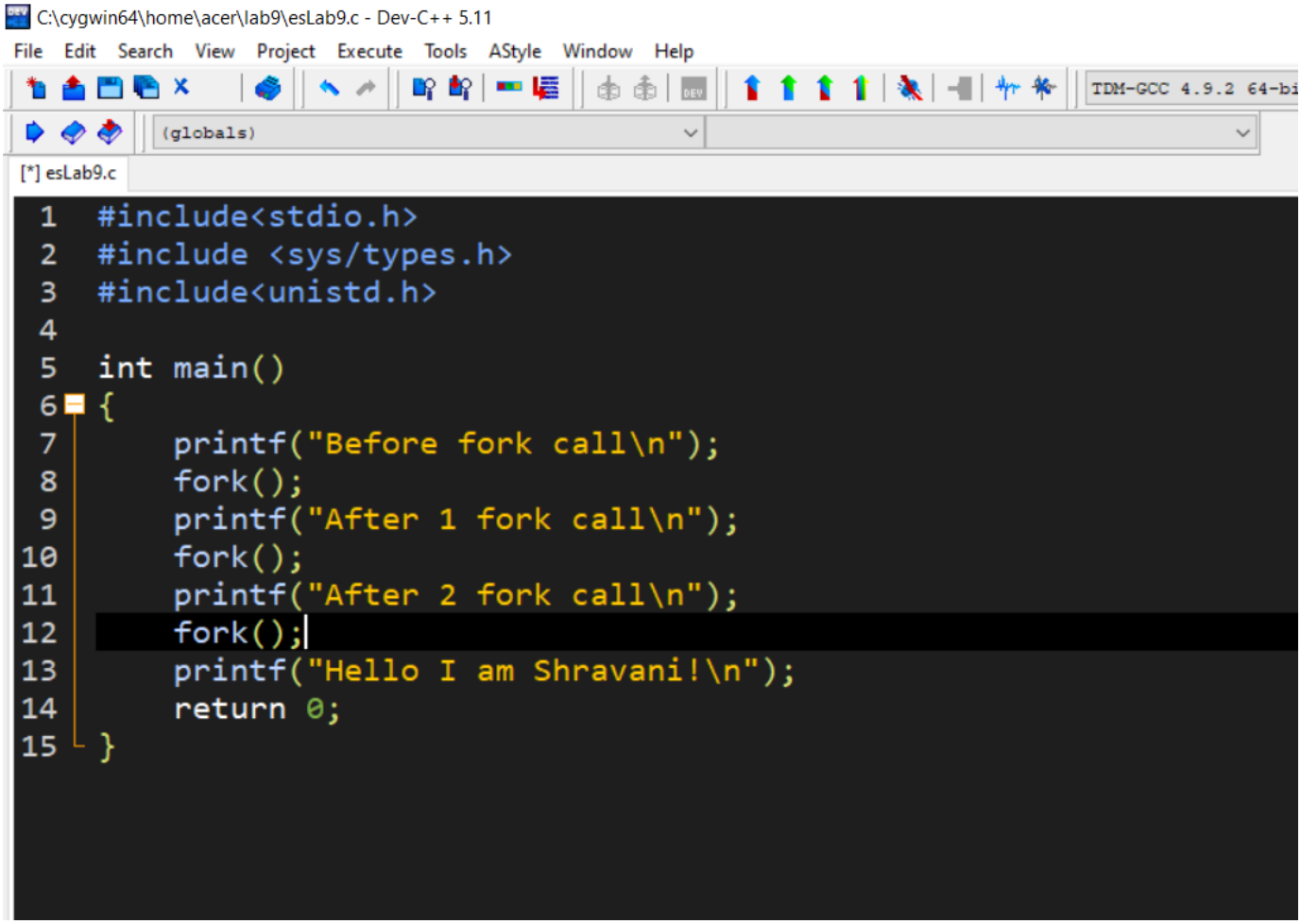
Friday, October 16, 2020 12:00 PM

Aim: **Program to introduce process and child process: introduction to fork system call, printing parent and child process IDs.**

Date: 16-10-20

Source code:

Understanding Child and parent using fork



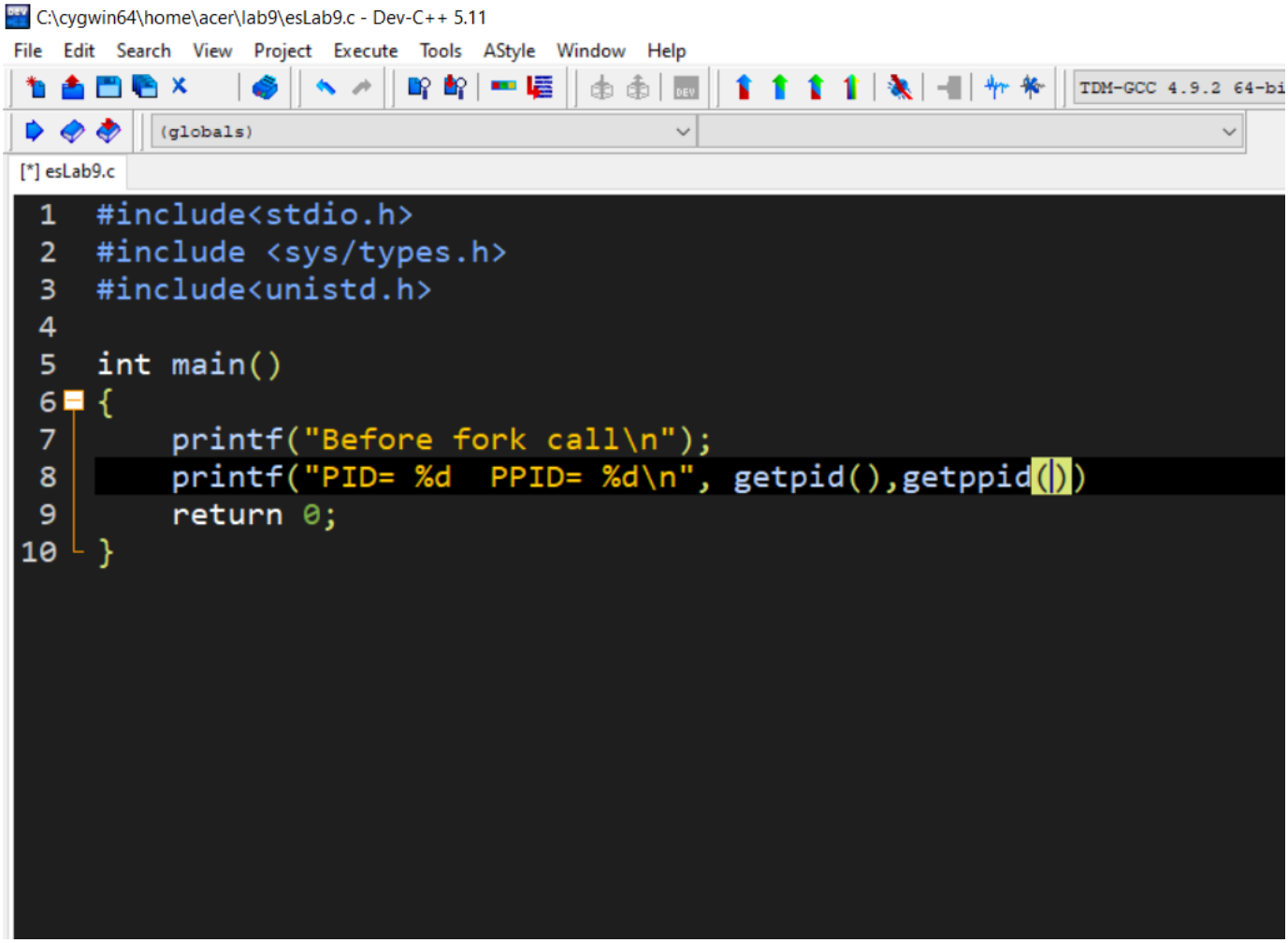
There are 6 printf statements printing because there are 3 forks under parent process

```
$ ./esLab9
Before fork call
Hello I am Shravani!
Hello I am Shravani!
Hello I am Shravani!
Hello I am Shravani!

acer@Shravani ~/lab9
$ gcc esLab9.c -o esLab9

acer@Shravani ~/lab9
$ ./esLab9
Before fork call
After 1 fork call
After 1 fork call
After 2 fork call
After 2 fork call
After 2 fork call
After 2 fork call
Hello I am Shravani!
Hello I am Shravani!
Hello I am Shravani!
Hello I am Shravani!
Hello I am Shravani!
Hello I am Shravani!
Hello I am Shravani!
Hello I am Shravani!
```

Printing PPID and PID



Always a different PPID and PID gets printed due to the system process

```
acer@Shravani ~/lab9
$ gcc esLab9.c -o esLab9

acer@Shravani ~/lab9
$ ./esLab9
Before fork call
PID =1879 PPID =1834
```

Checking with If else condition for Parent and child process:

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

int main()
{
    if (fork()==0){
        printf("Hello from child!\n");
    }
    else{
        printf("Hello from Parent!\n");
    }
    return 0;
}
```

Compiled output:

```
acer@Shravani ~/lab9
$ gcc esLab9.c -o esLab9

acer@Shravani ~/lab9
$ ./esLab9
Hello from child!
Hello from Parent!

acer@Shravani ~/lab9
$
```

Printing PPID and PID for parent and child

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

int main()
{
    if (fork()==0){
        printf("Hello from child!\n PID=%d PPID=%d\n", getpid(),
    }
    else{
        printf("Hello from Parent!\n PID=%d PPID=%d", getpid(),
    }
    return 0;
}
```

The output is different for PPID and PID of the Parent and Child

```
acer@Shravani ~/lab9
$ gcc esLab9.c -o esLab9

acer@Shravani ~/lab9
$ ./esLab9
Hello from Parent!
 PID=1892 PPID=1834Hello from child!
 PID=1893 PPID=1892
```

Inference:
In the above performed experiment we observed the function of parent and child using fork system call, we also printed the Parent IDs and Child Process IDs of them using getpid() and getppid() functions and obtained desired output.