OPERATING SYSTEMS LAB-3 WEEK-4

Author: Muhammad Ahsan 1912310 BSCS 5F

1. TASK 1: INSTALL C-LANGAUGE COMPILER IN LINUX OS

    -- sudo apt install gcc [INSTALLATION OF GCC FOR C]
    -- gcc --version [To Check Version of gcc compiler]


```
root@DESKTOP-HC9TP28:/mnt/g/OSLAB/oslab-1# gcc --version
gcc (Debian 8.3.0-6) 8.3.0
Copyright (C) 2018 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

2. TASK 2: WRITE A PROGRAM IN C TO PRINT "HELLO WORLD"

  -- nano hello.c [To Create C file to write hello world]

  -- #include <stdio.h>
    void main(){
     printf("Hello World");
    }

  -- CTRL+X [save quit to terminal and save file in nano]
  -- gcc hello.c -o hello [Compile C file with Output executable using this command]
  -- ./hello [Execute the file using this command]

  -- Now if we make changes we will just compile it and overwrite it with the same name
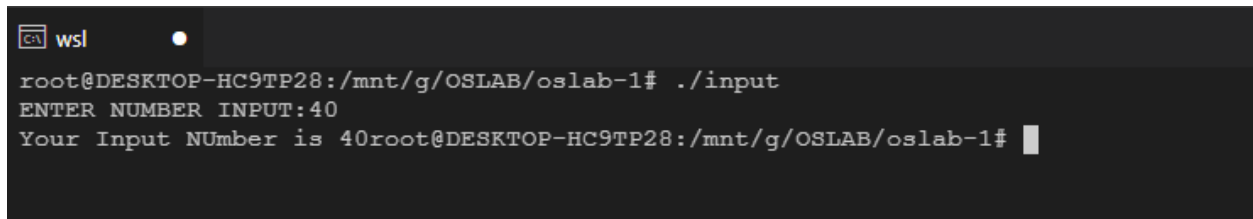 -- gcc hello.c -o hello
  -- ./hello


```
root@DESKTOP-HC9TP28:/mnt/g/OSLAB/oslab-1# ./hello
Hello Worldroot@DESKTOP-HC9TP28:/mnt/g/OSLAB/oslab-1#
```

3. TASK 3: TAKE INPUT FROM USER AND PRINT IT ON THE SCREEN

-- nano input.c [To Create C file to take input from user and print it on screen]
-- #include <stdio.h>

```c
void main(){
int number;
printf("ENTER NUMBER INPUT:");
scanf("%d",&number);
printf("Your Input Number is %d",number);
}
```

-- CTRL+X [save quit to terminal and save file in nano]
-- gcc input.c -o input [Compile C file with Output executable using this command]
-- ./input [Execute the file using this command]


```
root@DESKTOP-HC9TP28:/mnt/g/OSLAB/oslab-1# ./input
ENTER NUMBER INPUT:40
Your Input NUmber is 40root@DESKTOP-HC9TP28:/mnt/g/OSLAB/oslab-1#
```

4. TASK 4: TAKE INPUT NUMBER FROM USER AND PRINT IT'S TABLE ON THE SCREEN (TILL 10)

-- nano table.c
   [To Create C file to take input number from user and print it's table (TILL 10) on screen]
-- #include <stdio.h>

```c
void main(){
int number;
printf("ENTER NUMBER INPUT:");
scanf("%d",&number);
for(int i=1;i<=10;i++){
 printf("%d*%d=%d",i,number,i*number);
}
}
```

-- CTRL+X [save quit to terminal and save file in nano]
-- gcc table.c -o table [Compile C file with Output executable using this command]
-- ./table [Execute the file using this command]

```
root@DESKTOP-HC9TP28:/mnt/g/OSLAB/oslab-1# ./table
ENTER NUMBER INPUT:2
1*2=2
2*2=4
3*2=6
4*2=8
5*2=10
6*2=12
7*2=14
8*2=16
9*2=18
10*2=20
```

5. TASK 5: CALL FORK IN A PROGRAM TO CREATE SUB PROCESS AND COMPARE IT'S ID WITH
PARENT AND CHILD

   -- nano forkProgram.c
   -- #include <stdio.h>
     #include <unistd.h>
     void main(){
     int output=fork();
     if(output==0){
      printf("\n Called From Child \n");
     }
     else{
      printf(" \n Called From Parent");
     }
     }

   -- CTRL+X [save quit to terminal and save file in nano]
   -- gcc forkProgram.c -o forkProgram [Compile C file with Output executable using this
command]
   -- ./forkProgram [Execute the file using this command]

```
root@DESKTOP-HC9TP28:/mnt/g/OSLAB/oslab-1# ./forkProgram

 Called from parent

 Called from child
root@DESKTOP-HC9TP28:/mnt/g/OSLAB/oslab-1#
```
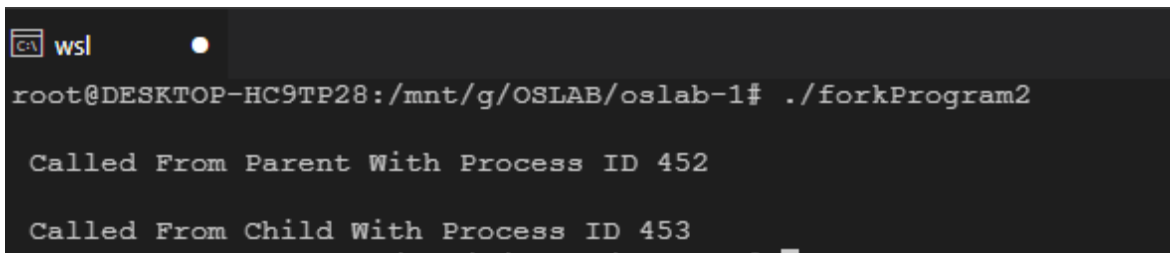
6. TASK 6: CALL FORK IN A PROGRAM TO CREATE SUB PROCESS AND COMPARE IT'S ID WITH PARENT AND CHILD AND OUTPUT IT'S PROCESS ID

-- nano forkProgram2.c
-- #include <stdio.h>
   #include <unistd.h>
   pid_t pid;

   void main(){
   int output=fork();
   pid=getpid();
   if(output==0){
    printf("\n Called From Child With Process ID %d \n",pid);
   }
   else{
    printf("\n Called From Parent With Process ID %d \n",pid);
   }
   }

-- CTRL+X [save quit to terminal and save file in nano]
-- gcc forkProgram2.c -o forkProgram2 [Compile C file with Output executable using this command]
-- ./forkProgram2 [Execute the file using this command]

```
CA wsl          ●
root@DESKTOP-HC9TP28:/mnt/g/OSLAB/oslab-1# ./forkProgram2

 Called From Parent With Process ID 452

 Called From Child With Process ID 453
```
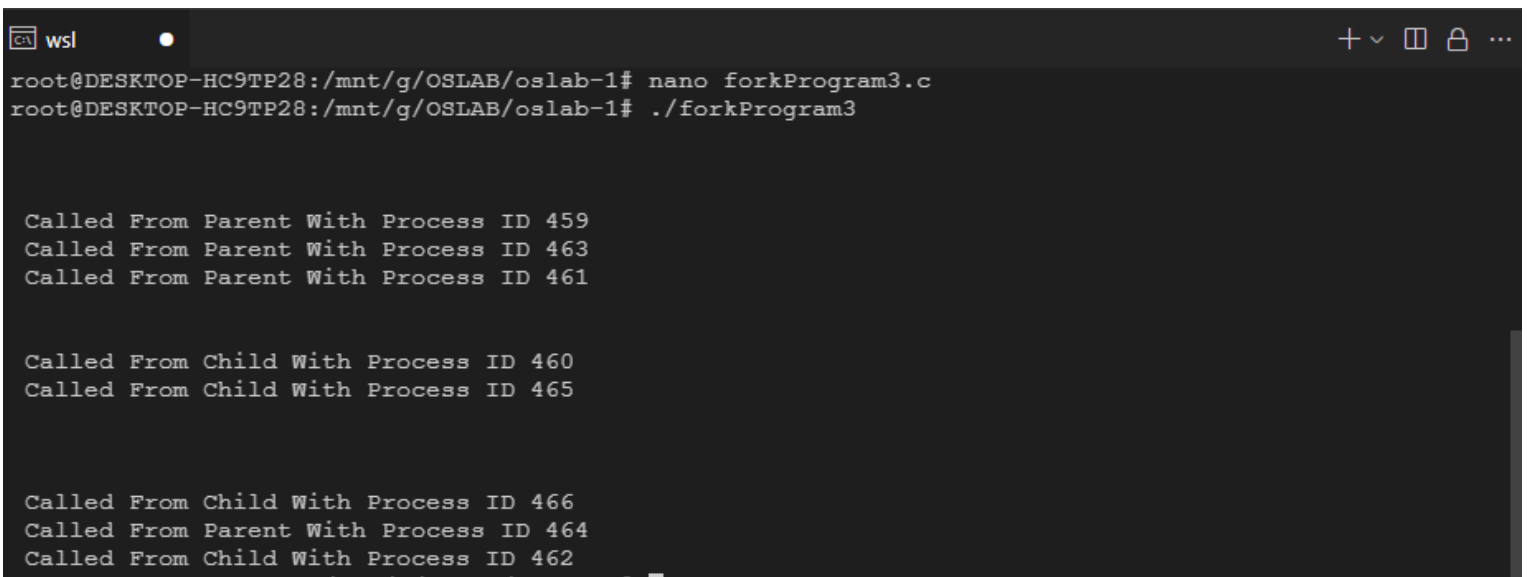
7. TASK 7: CALL FORK IN A PROGRAM TO CREATE SUB PROCESS (2 FORK) AND COMPARE IT'S ID
WITH PARENT AND CHILD AND OUTPUT IT'S PROCESS ID

-- nano forkProgram3.c
-- #include <stdio.h>
   #include <unistd.h>
   pid_t pid;

```c
void main(){
int output=fork();
fork();
fork();
pid=getpid();
if(output==0){
 printf("\n Called From Child With Process ID %d \n",pid);
}
else{
 printf("\n Called From Parent With Process ID %d \n",pid);
}
}
```

-- CTRL+X [save quit to terminal and save file in nano]
-- gcc forkProgram3.c -o forkProgram3 [Compile C file with Output executable using this
command]
-- ./forkProgram3 [Execute the file using this command]

```
[C:\] wsl                                                          + ∨  ⬚  🔒  ⋯
root@DESKTOP-HC9TP28:/mnt/g/OSLAB/oslab-1# nano forkProgram3.c
root@DESKTOP-HC9TP28:/mnt/g/OSLAB/oslab-1# ./forkProgram3


 Called From Parent With Process ID 459
 Called From Parent With Process ID 463
 Called From Parent With Process ID 461


 Called From Child With Process ID 460
 Called From Child With Process ID 465


 Called From Child With Process ID 466
 Called From Parent With Process ID 464
 Called From Child With Process ID 462
```