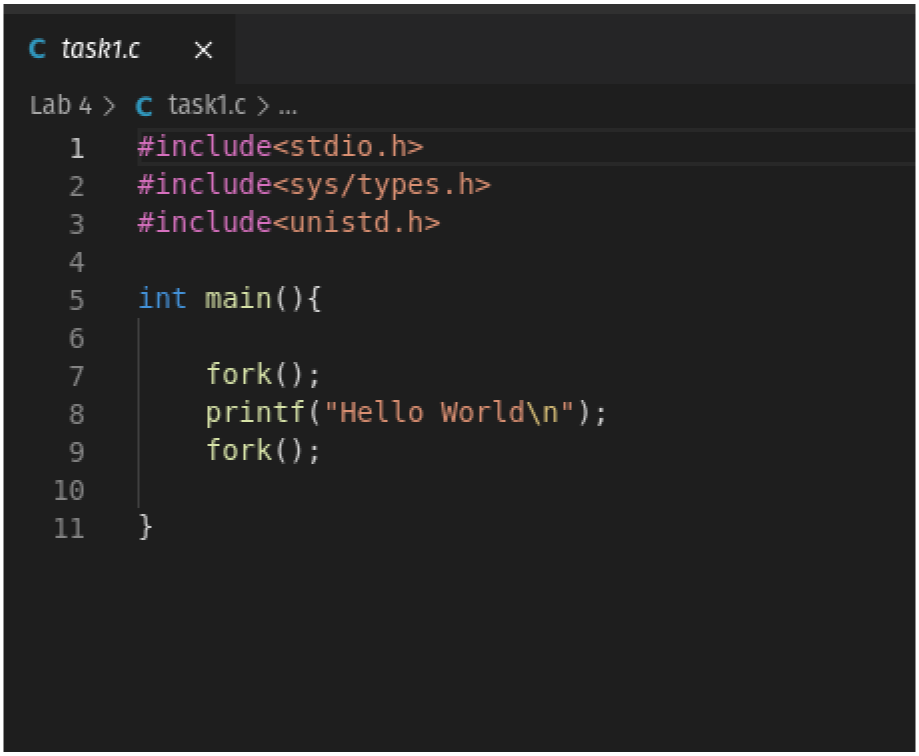
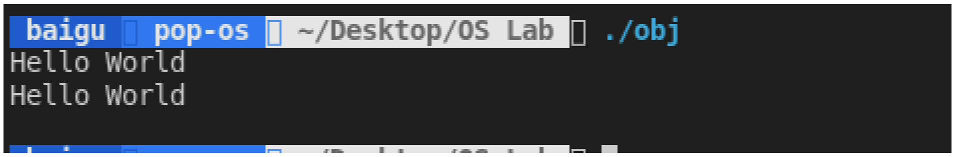
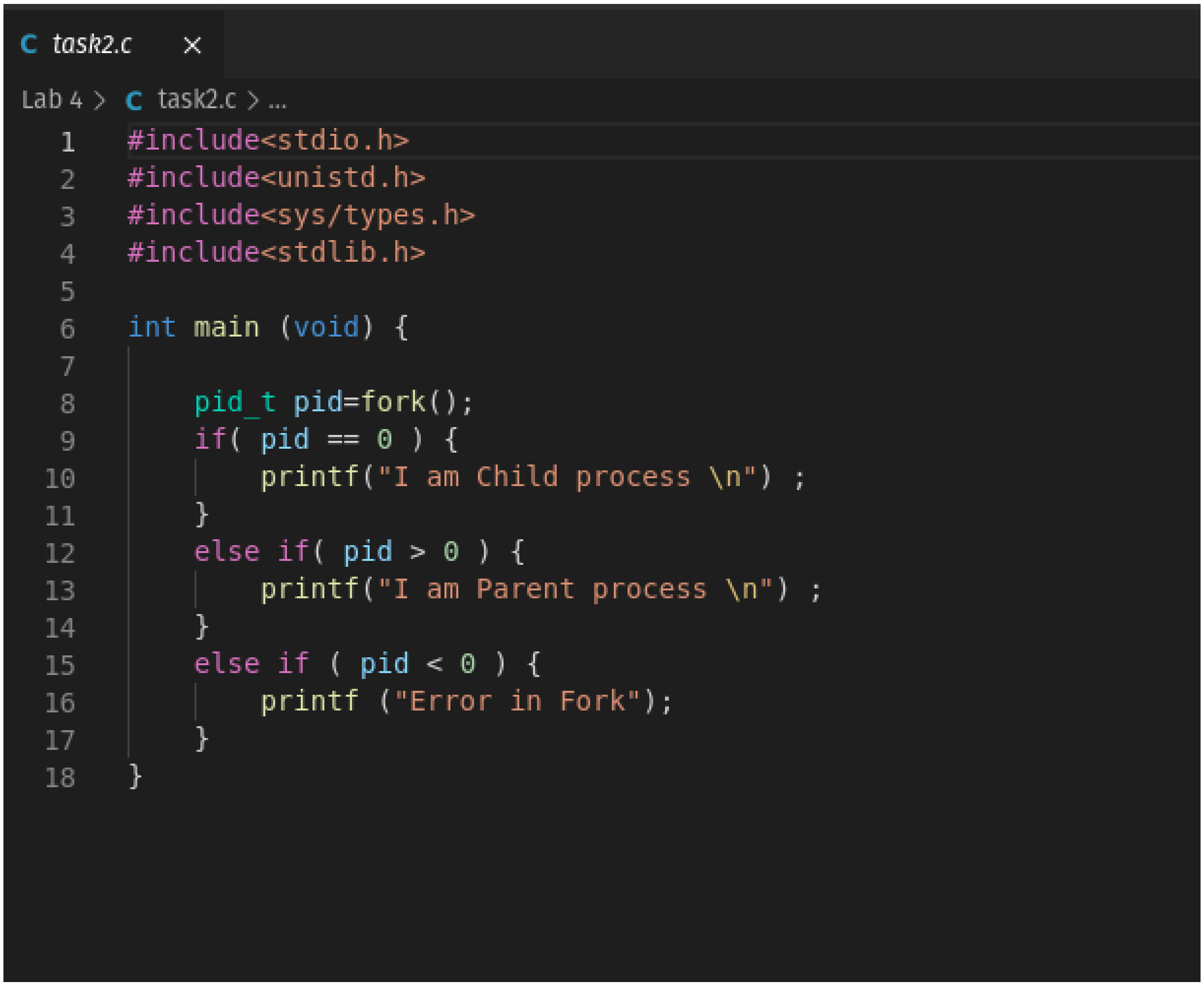


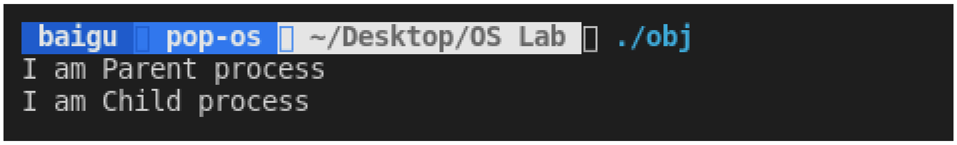
```c
#include<stdio.h>
#include<sys/types.h>
#include<unistd.h>

int main(){

    fork();
    printf("Hello World\n");
    fork();

}
```



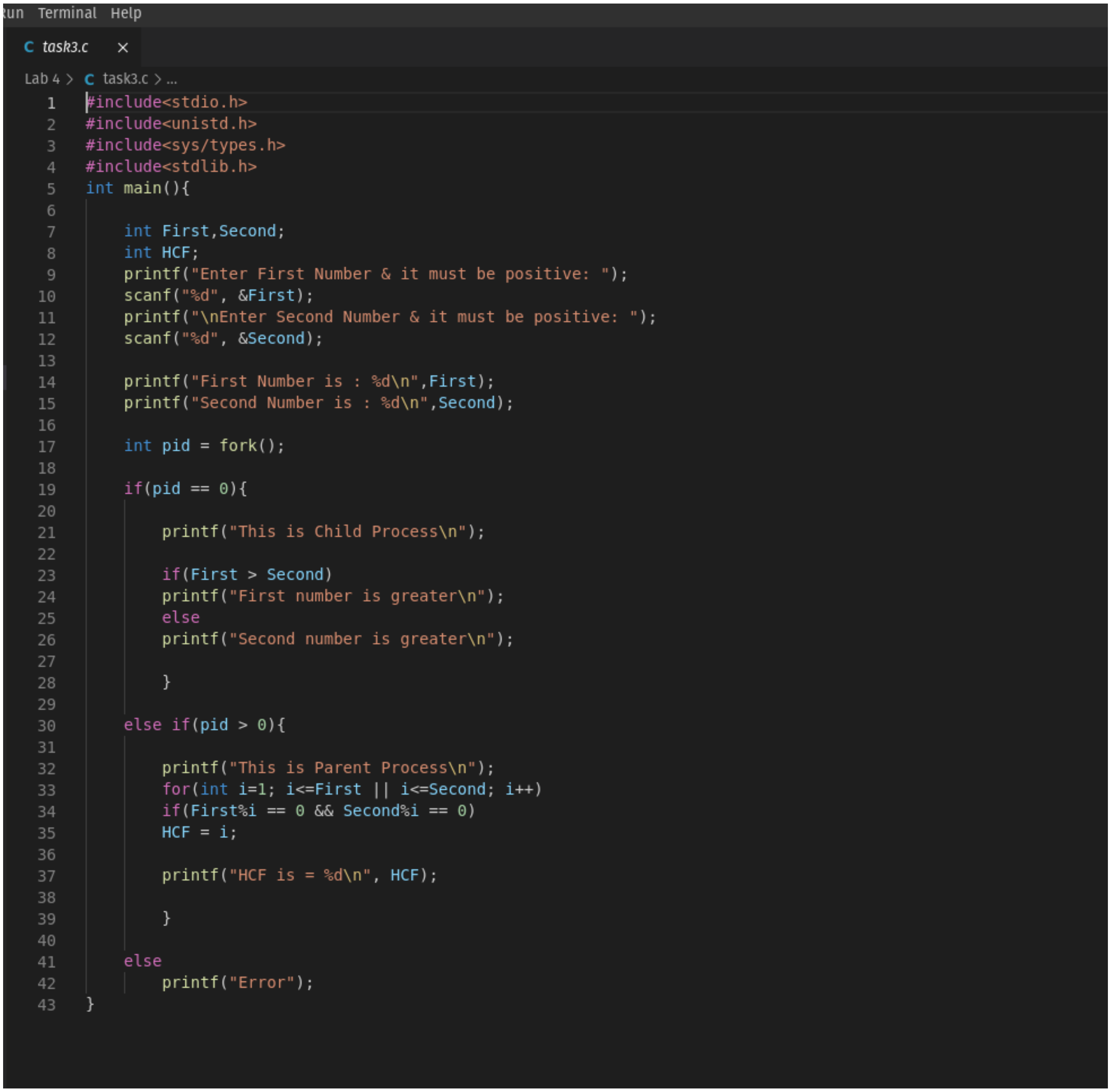
```
 baigu  pop-os  ~/Desktop/OS Lab  ./obj
Hello World
Hello World
```



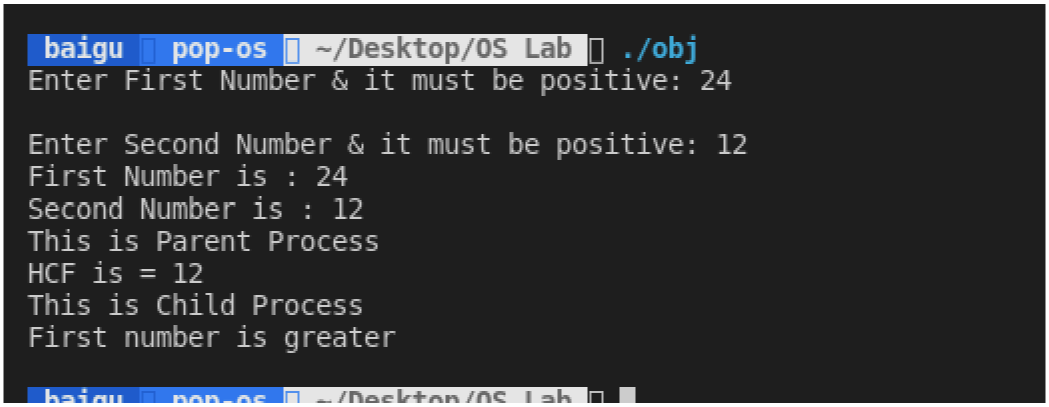
```c
#include<stdio.h>
#include<unistd.h>
#include<sys/types.h>
#include<stdlib.h>

int main (void) {

    pid_t pid=fork();
    if( pid == 0 ) {
        printf("I am Child process \n") ;
    }
    else if( pid > 0 ) {
        printf("I am Parent process \n") ;
    }
    else if ( pid < 0 ) {
        printf ("Error in Fork");
    }
}
```



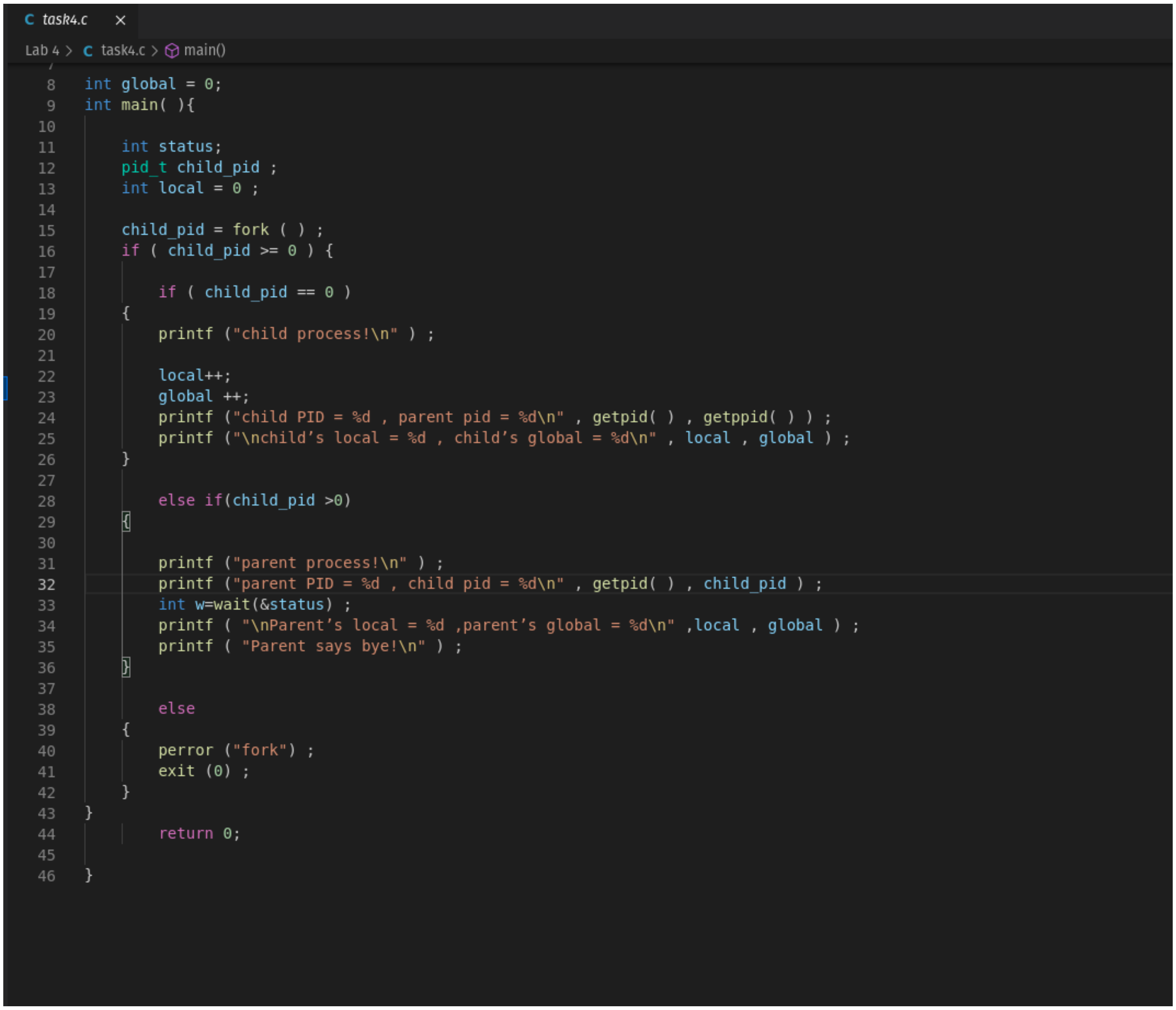
```
 baigu   pop-os   ~/Desktop/OS Lab   ./obj
I am Parent process
I am Child process
```



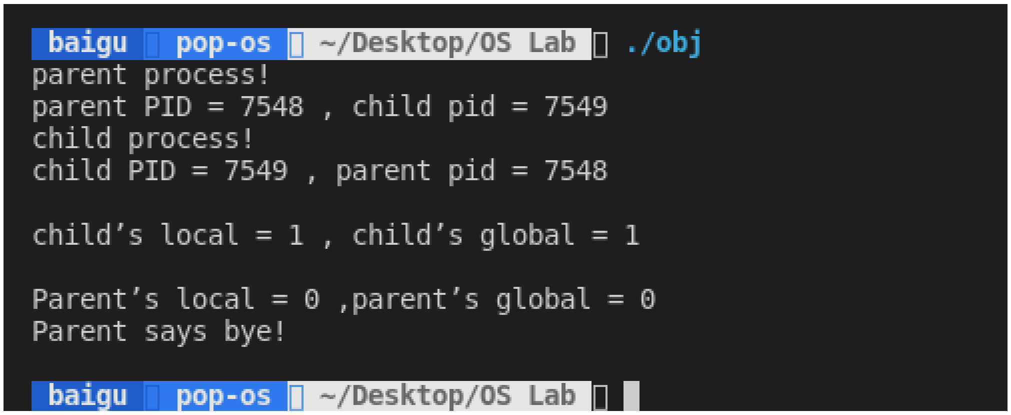
```c
#include<stdio.h>
#include<unistd.h>
#include<sys/types.h>
#include<stdlib.h>
int main(){

    int First,Second;
    int HCF;
    printf("Enter First Number & it must be positive: ");
    scanf("%d", &First);
    printf("\nEnter Second Number & it must be positive: ");
    scanf("%d", &Second);

    printf("First Number is : %d\n",First);
    printf("Second Number is : %d\n",Second);

    int pid = fork();

    if(pid == 0){

        printf("This is Child Process\n");

        if(First > Second)
        printf("First number is greater\n");
        else
        printf("Second number is greater\n");

        }

    else if(pid > 0){

        printf("This is Parent Process\n");
        for(int i=1; i<=First || i<=Second; i++)
        if(First%i == 0 && Second%i == 0)
        HCF = i;

        printf("HCF is = %d\n", HCF);

        }

    else
        printf("Error");
}
```



```
 baigu  pop-os  ~/Desktop/OS Lab  ./obj
Enter First Number & it must be positive: 24

Enter Second Number & it must be positive: 12
First Number is : 24
Second Number is : 12
This is Parent Process
HCF is = 12
This is Child Process
First number is greater

 baigu  pop-os  ~/Desktop/OS Lab 
```



```c
int global = 0;
int main( ){

    int status;
    pid_t child_pid ;
    int local = 0 ;

    child_pid = fork ( ) ;
    if ( child_pid >= 0 ) {

        if ( child_pid == 0 )
    {
        printf ("child process!\n" ) ;

        local++;
        global ++;
        printf ("child PID = %d , parent pid = %d\n" , getpid( ) , getppid( ) ) ;
        printf ("\nchild’s local = %d , child’s global = %d\n" , local , global ) ;
    }

        else if(child_pid >0)
    {

        printf ("parent process!\n" ) ;
        printf ("parent PID = %d , child pid = %d\n" , getpid( ) , child_pid ) ;
        int w=wait(&status) ;
        printf ( "\nParent’s local = %d ,parent’s global = %d\n" ,local , global ) ;
        printf ( "Parent says bye!\n" ) ;
    }

        else
    {
        perror ("fork") ;
        exit (0) ;
    }
}
        return 0;

}
```



```
baigu  pop-os  ~/Desktop/OS Lab  ./obj
parent process!
parent PID = 7548 , child pid = 7549
child process!
child PID = 7549 , parent pid = 7548

child’s local = 1 , child’s global = 1

Parent’s local = 0 ,parent’s global = 0
Parent says bye!

baigu  pop-os  ~/Desktop/OS Lab 
```