**JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY**

**WAKNAGHAT**

**Operating Systems Lab**

**Lab Assignment dated 04.02.2022**

**Name -** Akash Kumar Singh

**Roll no -** 201460

**Batch -** CS 48

**Task 1: WAP to create an orphan process with the following sample output.**

#include<stdio.h>

int main()

{

int x=fork();

if(x==0)

{

sleep(5);

printf("I am the child process with PID %d.\n",getpid());

printf("My Parent has terminated.\n");

printf("I am an orphan process now.\n");

printf("My Parent's process ID is %d.\n",getppid());

}

else

{

printf("I am the parent process with PID %d.\n",getpid());

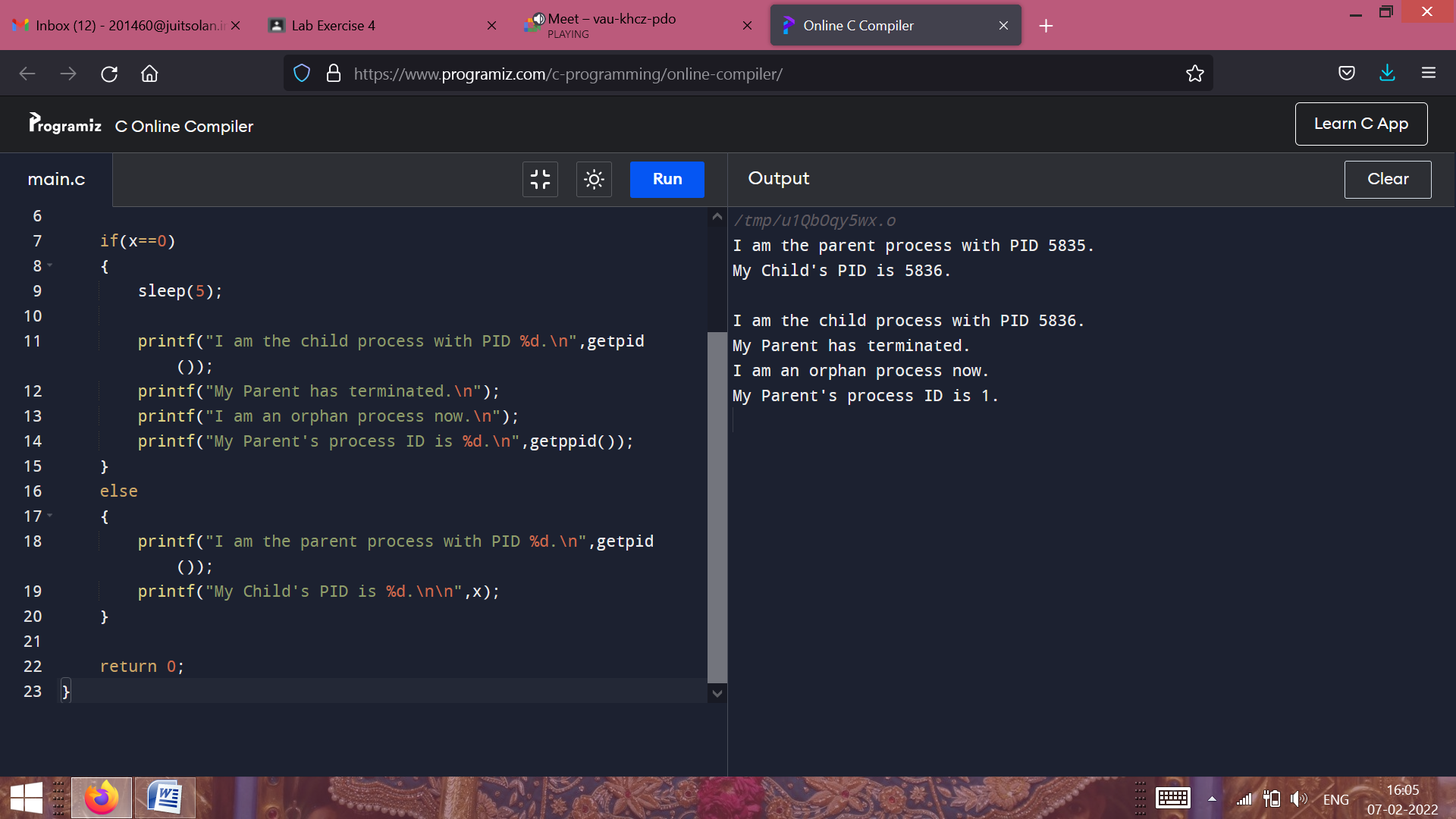
printf("My Child's PID is %d.\n\n",x);

}

return 0;

}

**//Output –**

****

**Task 2: WAP to create a zombie process in the system using any logic/method.**

#include<stdio.h>

#include<unistd.h>

#include<sys/wait.h>

#include<sys/types.h>

int main()

{

int x=fork();

if(x==0)

{

printf("I am the child process with PID %d.\n",getpid());

printf("My Parent's process ID is %d.\n",getppid());

}

else

{

int pid,status;

pid=wait(&status);

printf("%d\n",pid);

printf("Status of the child process is %d\n",WIFEXITED(status));

//If the status of child process is 1, it means it has terminated

printf("There exists a Zombie Process\n");

printf("I am the parent process with PID %d.\n",getpid());

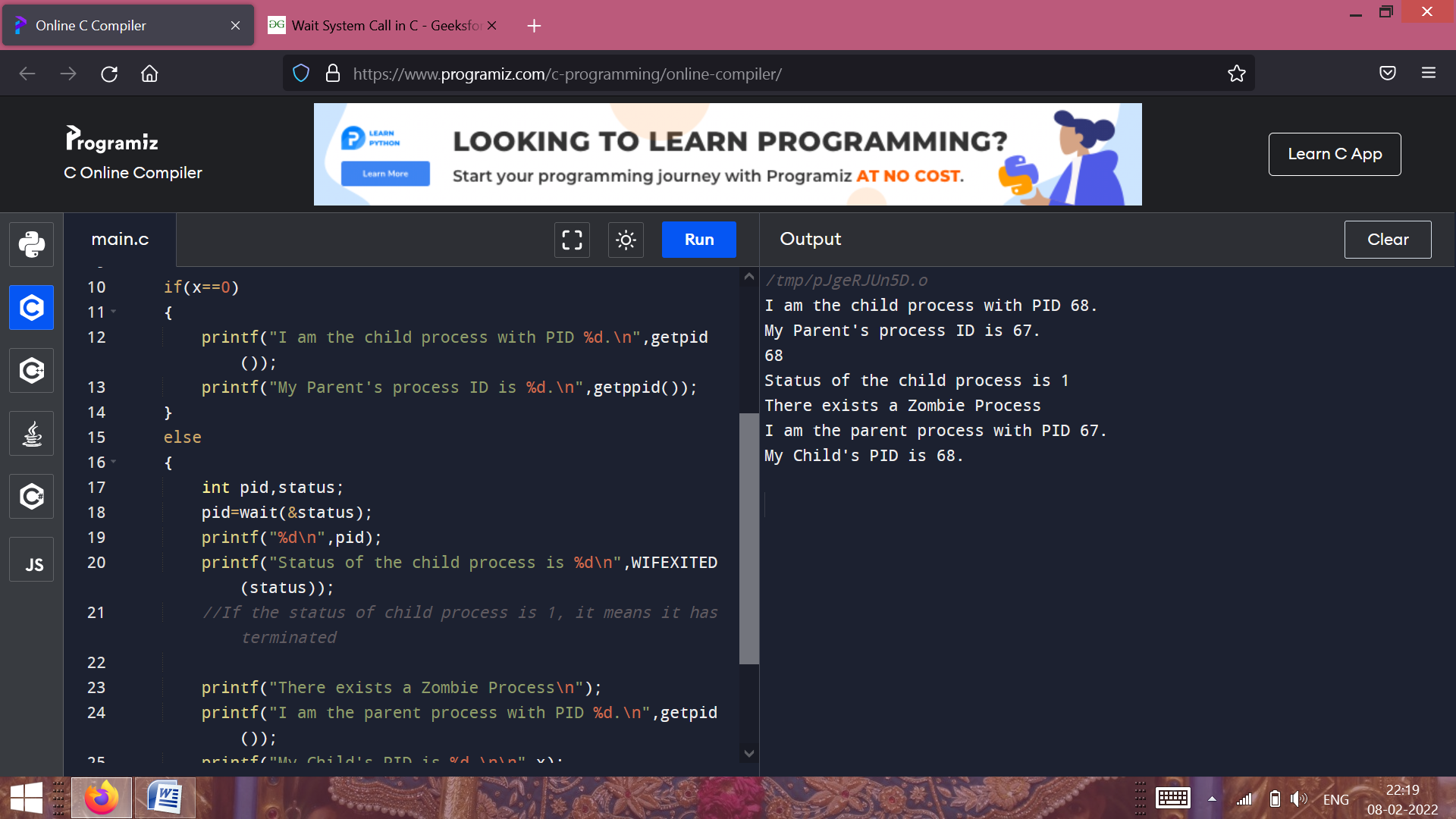
printf("My Child's PID is %d.\n\n",x);

}

return 0;

}

**//Output –**

****