The exec Function

LAB # 04

Fall 2024

CSE-302L

Systems Programming Lab

Submitted by: **AIMAL KHAN**

Registration No.: **21PWCSE1996**

Class Section: **A**

"On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work."

Student Signature: _______________

Submitted to:

Engr. Abdullah Hamid

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar

CSE 302L: SYSTEMS PROGRAMMING LAB

LAB ASSESSMENT RUBRICS

Criteria & Point Assigned	**Outstanding 2**	**Acceptable 1.5**	**Considerable 1**	**Below Expectations 0.5**	**Score**
Attendance and Attentiveness in Lab PLO08	Attended in proper Time and attentive in Lab	Attended in proper Time but not attentive in Lab	Attended late but attentive in Lab	Attended late not attentive in Lab	
Capability of writing Program/Algorithm/Drawing Flow Chart PLO1, PLO2, PLO3, PLO5	Right attempt/ no errors and well formatted	Right attempt/ no errors but not well formatted	Right attempt/ minor errors and not well formatted	Wrong attempt	
Result or Output/ Completion of target in Lab PLO9	100% target has been completed and well formatted.	75% target has been completed and well formatted.	50% target has been completed but not well formatted.	None of the outputs are correct.	
Overall, Knowledge PLO10,	Demonstrates excellent knowledge of lab	Demonstrates good knowledge of lab	Has partial idea about the Lab and procedure followed	Has poor idea about the Lab and procedure followed	
Attention to Lab Report PLO4,	Submission of Lab Report in Proper Time i.e., in next day of lab, with proper documentation.	Submission of Lab Report in proper time but not with proper documentation.	Late Submission with proper documentation.	Late Submission very poor documentation	

Instructor:

Name: ___________________________	Signature: __________________________

The exec Function

Objectives:

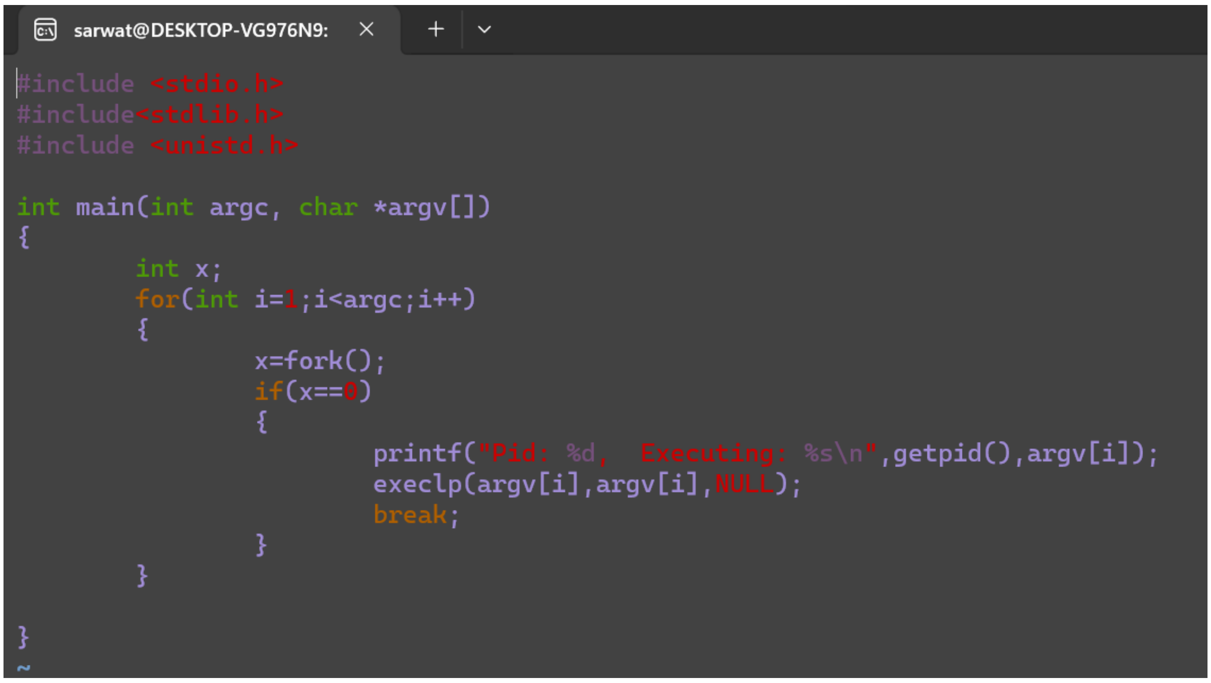
In this lab we will learn about the exec function.

Tasks:

Task 1:

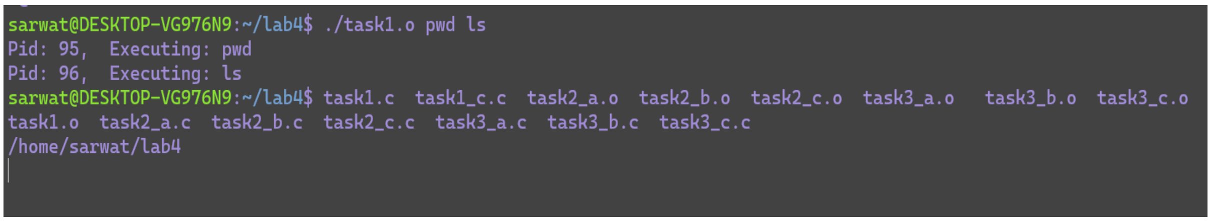
Write a program that takes N UNIX commands as arguments, creates N child processes, each of them implementing their respective commands. Parent process shall wait for all the child processes and receive and print the exit status of the child processes.

Code in C:



```c
#include <stdio.h>
#include<stdlib.h>
#include <unistd.h>

int main(int argc, char *argv[])
{
        int x;
        for(int i=1;i<argc;i++)
        {
                x=fork();
                if(x==0)
                {
                        printf("Pid: %d,  Executing: %s\n",getpid(),argv[i]);
                        execlp(argv[i],argv[i],NULL);
                        break;
                }
        }

}
~
```

Output:

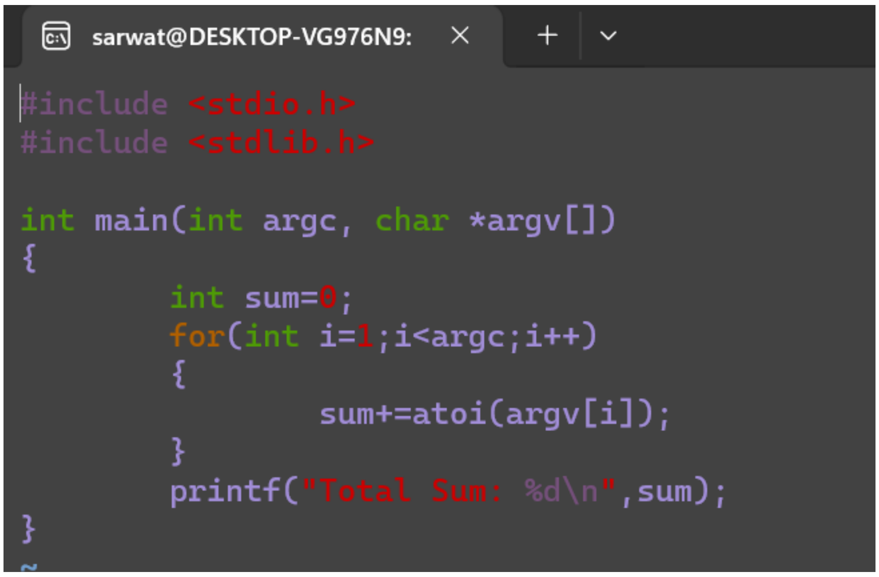


```
sarwat@DESKTOP-VG976N9:~/lab4$ ./task1.o pwd ls
Pid: 95,  Executing: pwd
Pid: 96,  Executing: ls
sarwat@DESKTOP-VG976N9:~/lab4$ task1.c  task1_c.c  task2_a.o  task2_b.o  task2_c.o  task3_a.o   task3_b.o  task3_c.o
task1.o  task2_a.c  task2_b.c  task2_c.c  task3_a.c  task3_b.c  task3_c.c
/home/sarwat/lab4
```

Task 2:

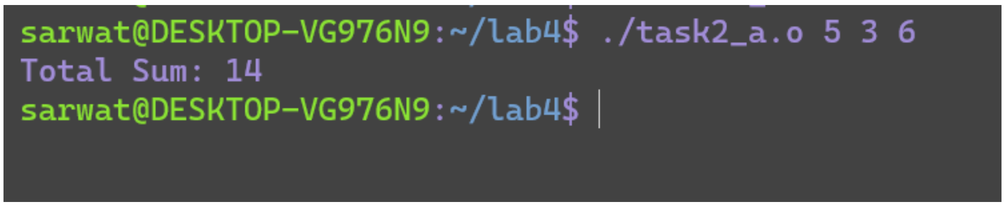
a. Write a program that takes integers as arguments and adds them. b. Write a program that takes integers as arguments and multiplies them. c. Write a program that takes integers as arguments & adds & multiplies them using the above two programs.

Part a:



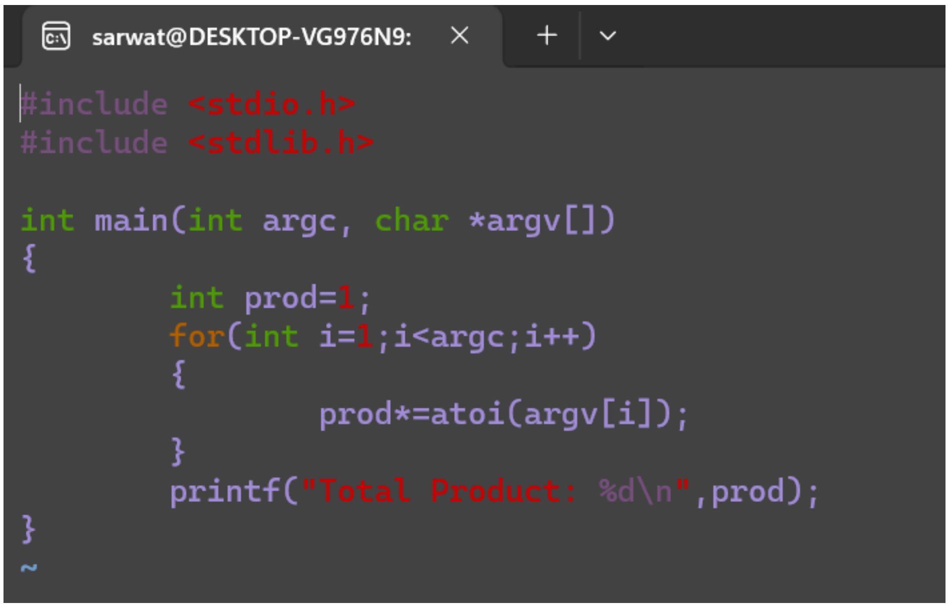
```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
        int sum=0;
        for(int i=1;i<argc;i++)
        {
                sum+=atoi(argv[i]);
        }
        printf("Total Sum: %d\n",sum);
}
```

Output:



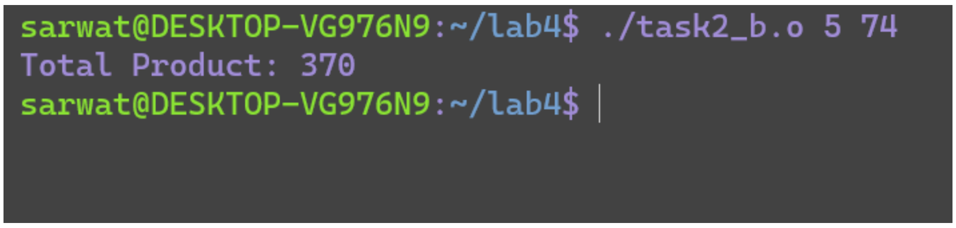
```
sarwat@DESKTOP-VG976N9:~/lab4$ ./task2_a.o 5 3 6
Total Sum: 14
sarwat@DESKTOP-VG976N9:~/lab4$
```

Part b:



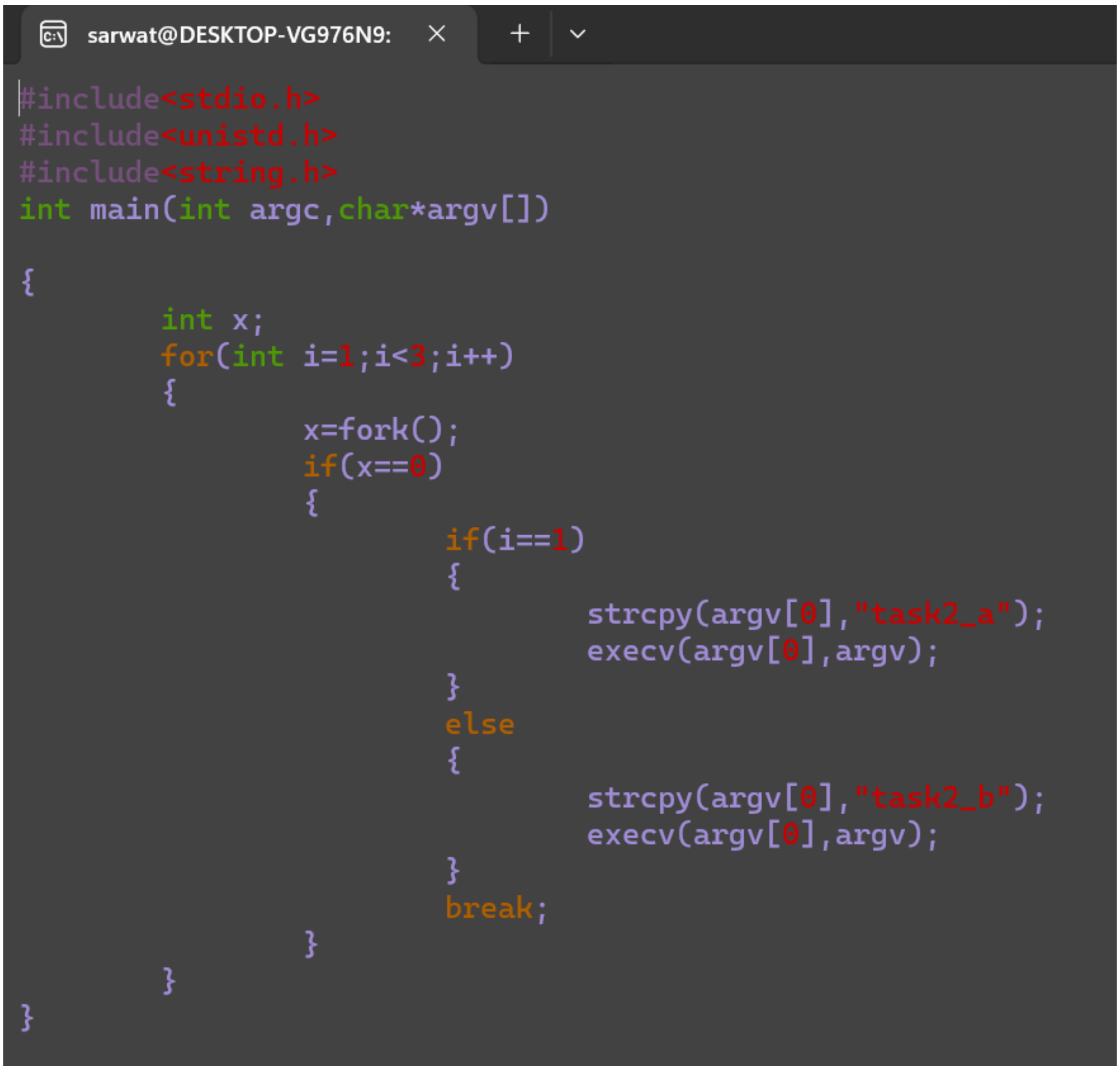
```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
        int prod=1;
        for(int i=1;i<argc;i++)
        {
                prod*=atoi(argv[i]);
        }
        printf("Total Product: %d\n",prod);
}
~
```

Output:



```
sarwat@DESKTOP-VG976N9:~/lab4$ ./task2_b.o 5 74
Total Product: 370
sarwat@DESKTOP-VG976N9:~/lab4$
```

Part c:

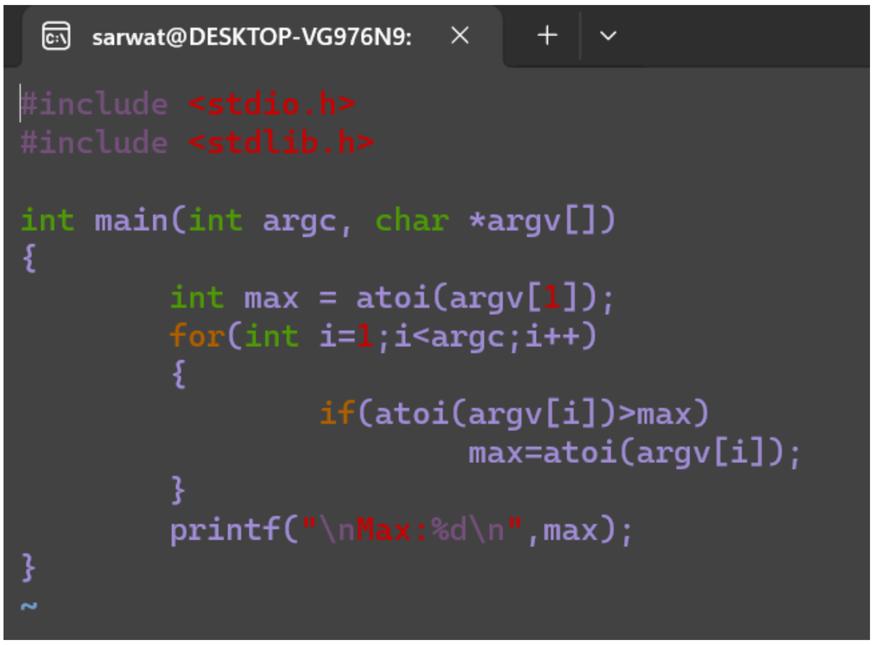


```
#include<stdio.h>
#include<unistd.h>
#include<string.h>
int main(int argc,char*argv[])

{
        int x;
        for(int i=1;i<3;i++)
        {
                x=fork();
                if(x==0)
                {
                        if(i==1)
                        {
                                strcpy(argv[0],"task2_a");
                                execv(argv[0],argv);
                        }
                        else
                        {
                                strcpy(argv[0],"task2_b");
                                execv(argv[0],argv);
                        }
                        break;
                }
        }
}
```

Task 3

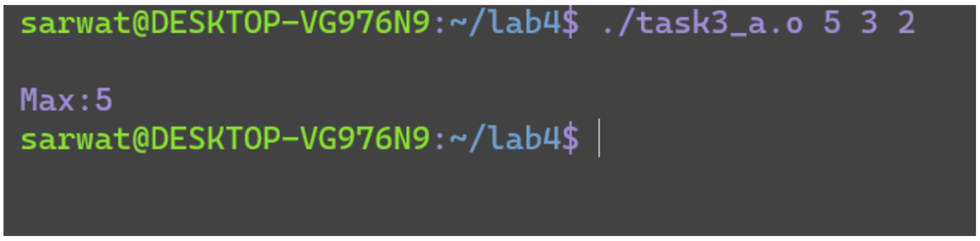
Write a program “minmax.c” that takes an array as command line arguments. Program executes min.c and max.c programs in its two child processes. One child process calculates and returns the min value and other calculates and returns the max value in the array. The program “minmax.c” shall receive the values returned by the child processes and display these values.

Part a:


sarwat@DESKTOP-VG976N9:

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
        int max = atoi(argv[1]);
        for(int i=1;i<argc;i++)
        {
                if(atoi(argv[i])>max)
                        max=atoi(argv[i]);
        }
        printf("\nMax:%d\n",max);
}
~
```

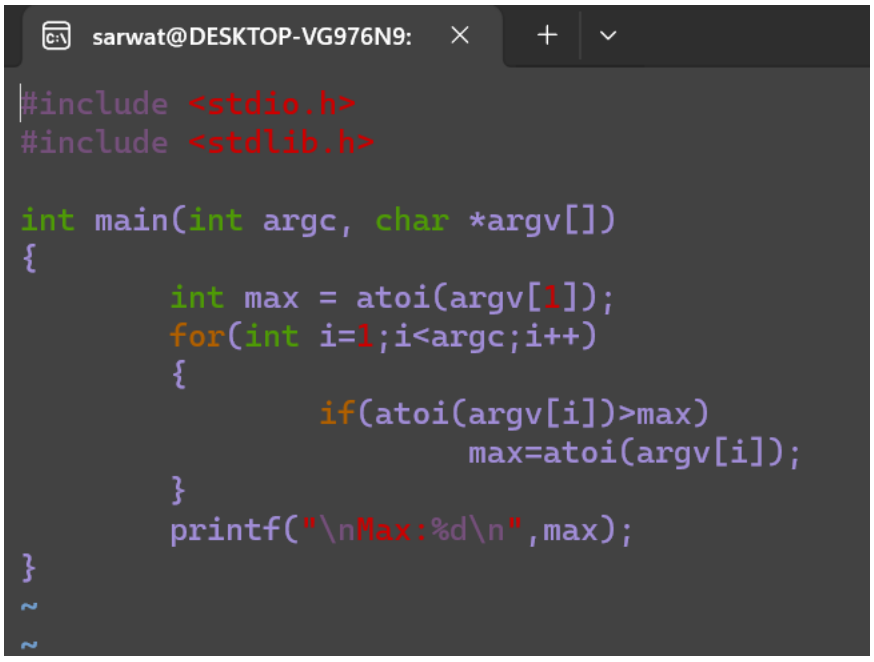
Output:



```
sarwat@DESKTOP-VG976N9:~/lab4$ ./task3_a.o 5 3 2

Max:5
sarwat@DESKTOP-VG976N9:~/lab4$
```

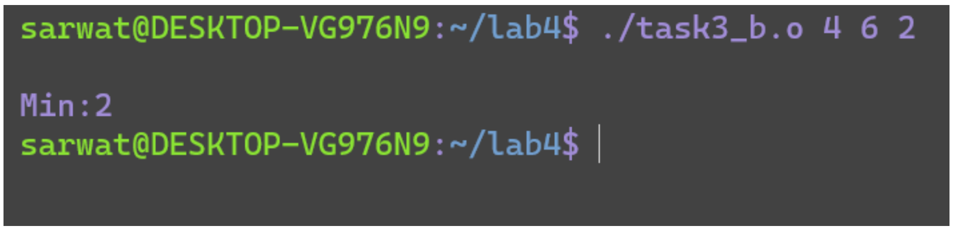
Part b:

Code in c:


sarwat@DESKTOP-VG976N9:

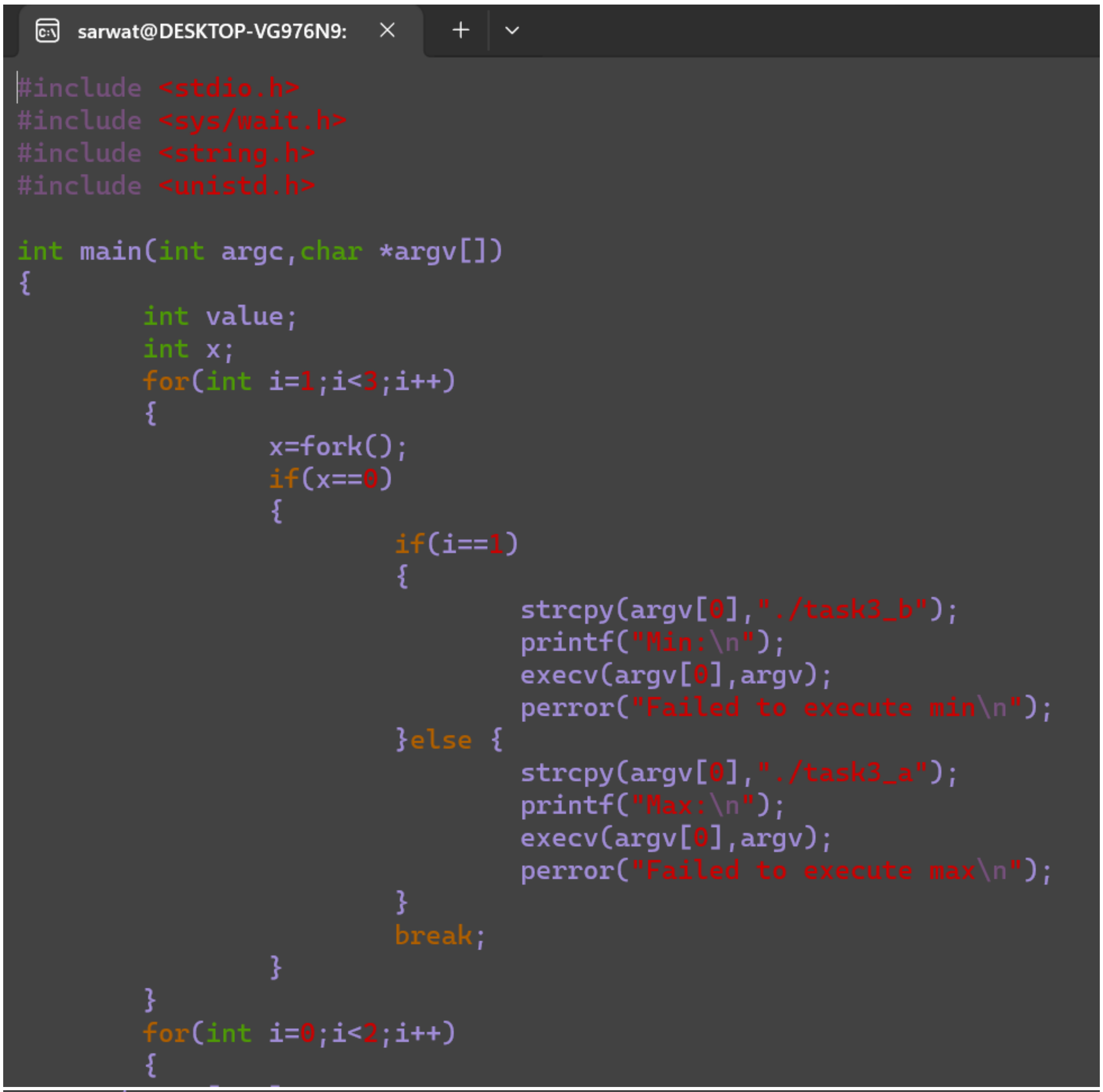
```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
        int max = atoi(argv[1]);
        for(int i=1;i<argc;i++)
        {
                if(atoi(argv[i])>max)
                        max=atoi(argv[i]);
        }
        printf("\nMax:%d\n",max);
}
~
~
```

Output:

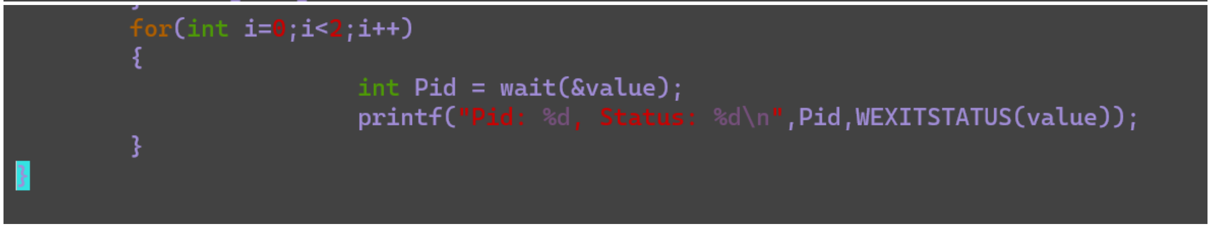


```
sarwat@DESKTOP-VG976N9:~/lab4$ ./task3_b.o 4 6 2

Min:2
sarwat@DESKTOP-VG976N9:~/lab4$
```

Part c:



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

int main(int argc,char *argv[])
{
        int value;
        int x;
        for(int i=1;i<3;i++)
        {
                x=fork();
                if(x==0)
                {
                        if(i==1)
                        {
                                strcpy(argv[0],"./task3_b");
                                printf("Min:\n");
                                execv(argv[0],argv);
                                perror("Failed to execute min\n");
                        }else {
                                strcpy(argv[0],"./task3_a");
                                printf("Max:\n");
                                execv(argv[0],argv);
                                perror("Failed to execute max\n");
                        }
                        break;
                }
        }
        for(int i=0;i<2;i++)
        {
```



```c
        for(int i=0;i<2;i++)
        {
                int Pid = wait(&value);
                printf("Pid: %d, Status: %d\n",Pid,WEXITSTATUS(value));
        }
}
```

Conclusion:

In conclusion, I have learned about exec function.

The End.