**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÀI TẬP MÔN HỌC NMHĐH**

**LAB 6**

**Nhóm:** 22 **Tổ:2**

Họ và tên**: Nguyễn Thị Anh Thư** (MSSV: **51900564**)

**NỘI DUNG BÁO CÁO KẾT QUẢ**

**Câu 1A.** Tiến trình cha chuyển đối số đầu tiên (argv [1]) là một số nguyên lớn hơn 3 cho tiến trình con thông qua

đường ống. Tiến trình con nhận, tính giá trị n! = 1 \* 2 \* ... \* n và ghi nó vào đường ống. Tiến trình cha nhận

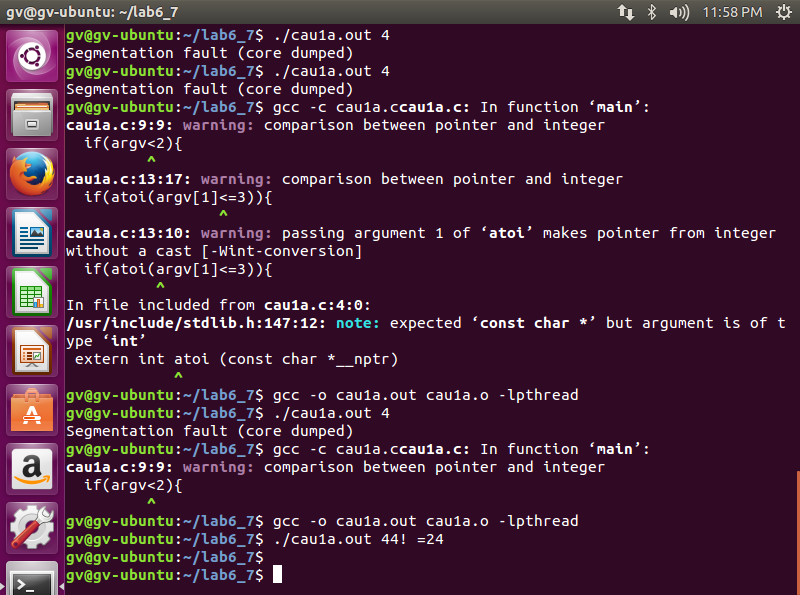
và xuất dữ liệu ra màn hình. Sử dụng đường ống vô danh (Unnamed Pipe).

**KẾT QUẢ THỰC HIỆN**

**1A.1. Phần source code:**

1. #include<stdio.h>
2. #include<unistd.h>
3. #include<string.h>
4. #include<stdlib.h>
5. int main(int argc, char\* argv[]){
6. int fp1[2],fp2[2];
7. int buffer;
8. int pid;
9. if(argv<2){
10. printf("Doi so thieu: \n ");
11. return -1;
12. }
13. if(atoi(argv[1])<=3){
14. printf("Doi so khong lon hon 3 \n");
15. return -1;
16. }
17. else{
18. if(pipe(fp1)==0 && pipe(fp2)==0){
19. pid=fork();
20. if( pid<0){
21. printf("Fork failed\n");
22. return -1;
23. }
24. else if(pid==0){
25. close(fp1[1]);
26. read(fp1[0],&buffer,sizeof(buffer));
27. close(fp1[0]);
28. int s=1,i;
29. for(i=2;i<=buffer;i++)
30. s=s\*i;
31. close(fp2[0]);
32. write(fp2[1],&s,sizeof(s));
33. close(fp2[1]);
34. }
35. else{
36. close(fp1[0]);
37. int temp = atoi(argv[1]);
38. write(fp1[1],&temp, sizeof(temp));
39. close(fp2[1]);
40. int n;
41. read(fp2[0],&n,sizeof(n));
42. printf("%d! =%d \n",temp,n);
43. close(fp2[0]);
44. }
45. }
46. else{
47. printf("pipe faile \n");
48. return -2;
49. }
50. }
51. return 1;
52. }

**1A.2. Kết quả chạy chương trình:**

******

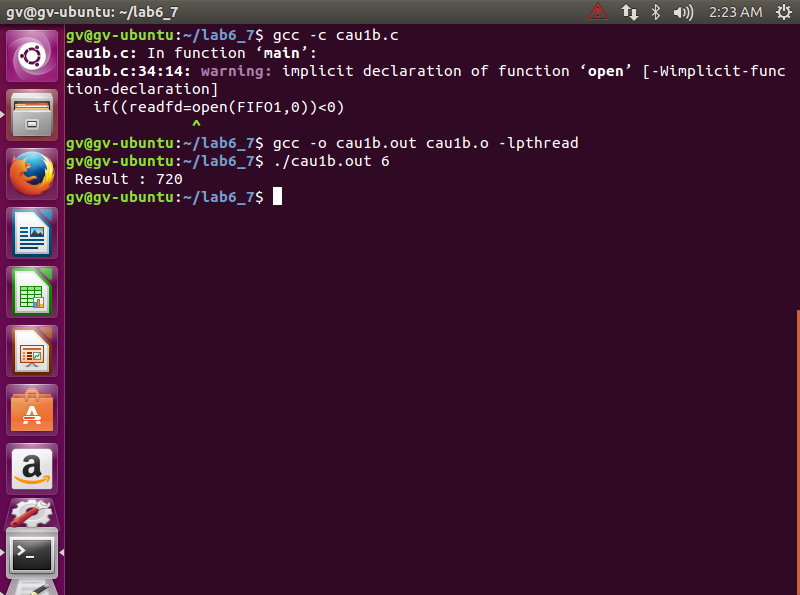
**Bài 1B.** Giải lại vấn đề 1A với đường ống có tên (Named Pipe).

**KẾT QUẢ THỰC HIỆN**

**1b.1 Phần source code:**

1. #include<stdio.h>
2. #include<stdlib.h>
3. #include<unistd.h>
4. #include<string.h>
5. #include<sys/types.h>
6. #include<sys/stat.h>
7. #include<sys/errno.h>
8. #define FIFO1 "/tmp/ff.1"
9. #define FIFO2 "/tmp/ff.2"
10. #define PM 0666
11. extern int errno;
12. int main(int n, char \*\*a){
13. if(n<2){
14. printf(" Missing Ardument\n");
15. return -1;
16. }
17. if(atoi(a[1])<=3){
18. printf("Doi so khong lon hon 3");
19. return -1;
20. }
21. else{
22. int childpid, readfd, writefd;
23. if((mknod(FIFO1,S\_IFIFO | PM,0)<0)&& (errno!=EEXIST)){
24. printf(" Fail to create FIFO 1.Aborted. \n");
25. return -1;
26. }
27. if((mknod(FIFO2,S\_IFIFO | PM,0)<0)&& (errno!=EEXIST)){
28. unlink(FIFO1);
29. printf(" Fail to create FIFO 2.Aborted. \n");
30. return -1;
31. }
32. childpid=fork();
33. if (childpid ==0){
34. if((readfd=open(FIFO1,0))<0)
35. perror(" child cannot open readFIFO. \n");
36. if((writefd=open(FIFO2,1))<0)
37. perror(" child cannot open writeFIFO. \n");
38. int s;
39. read (readfd,&s,sizeof(s));
40. int gt=1;
41. for ( int i=1; i<=s;i++){
42. gt=gt\*i;
43. }write(writefd,&gt,sizeof(gt));
44. close(readfd);
45. close(writefd);
46. return 1;
47. }
48. else if( childpid>0){
49. if((writefd=open(FIFO1,1))<0)
50. perror(" Child cannot open writeFIFO. \n");
51. if((readfd =open(FIFO2,0))<0)
52. perror(" Child cannot open raedFIFO. \n");
53. int input=atoi(a[1]);
54. write(writefd,&input,sizeof(input));
55. int temp;
56. read (readfd,&temp,sizeof(temp));
57. printf(" Result : %d\n",temp);
58. close(readfd);
59. close(writefd);
60. if ( unlink(FIFO1)<0)
61. perror(" cannot remove FIFO1. \n");
62. if(unlink(FIFO2)<0)
63. perror("Cannot remove FIDO2. \n");
64. return 1;
65. }
66. else{
67. printf(" Fork failed \n");
68. return -1;
69. }
70. }
71. }

**1b.2 Kết quả chạy chương trình:**

****

**Bài 2A.** Tiến trình cha đọc hai số nguyên và một thao tác +, -, \*, / và chuyển tất cả cho tiến trình con. Quá trình

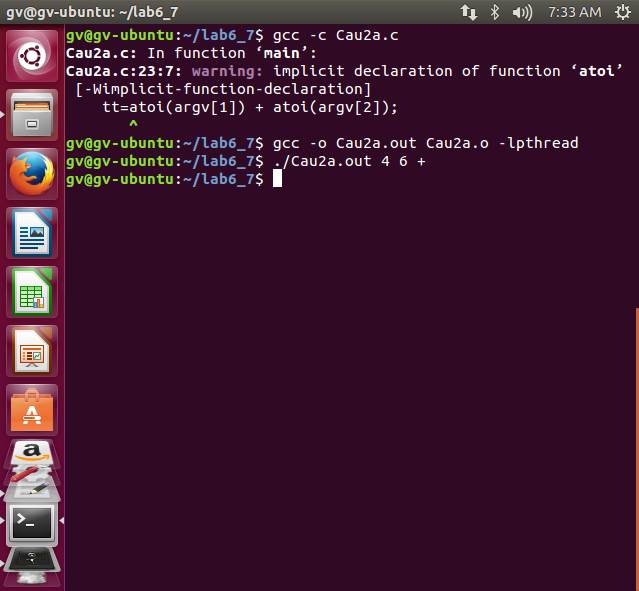
con tính toán kết quả và trả về cho tiến trình cha. Quá trình cha mẹ ghi kết quả vào một tệp**.**

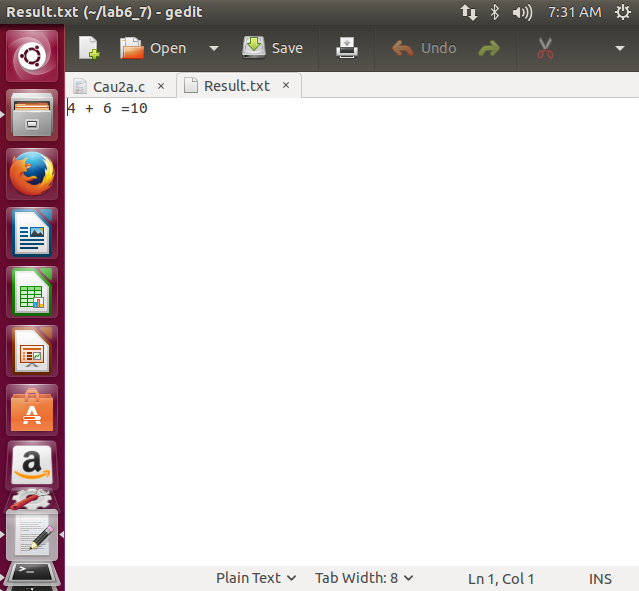
**KẾT QUẢ THỰC HIỆN**

**2A.1 Phần source code:**

1. #include<stdio.h>
2. #include<unistd.h>
3. #include<string.h>
4. int main( int argc, char\* argv[]){
5. int fp[2];
6. int pid;
7. int kq,tt;
8. if (argc<4){
9. printf(" Doi so khong dung\n");
10. return -1;
11. }
12. if(pipe(fp)==0){
13. pid=fork();
14. if(pid <0){
15. printf(" Fork failed\n");
16. return -1;
17. }else if(pid ==0){
18. int pt= \*argv[3];
19. int i;
20. if(pt == '+'){
21. tt=atoi(argv[1]) + atoi(argv[2]);
22. }
23. else if(pt == '-'){
24. tt=atoi(argv[1]) - atoi(argv[2]);
25. }
26. else if(pt == '\*'){
27. tt=atoi(argv[1]) \* atoi(argv[2]);
28. }
29. else if(pt == '/'){
30. tt=atoi(argv[1]) / atoi(argv[2]);
31. }
32. else{
33. tt=0;
34. }
35. close(fp[0]);
36. write(fp[1],&tt,sizeof(tt));
37. }
38. else{
39. close(fp[1]);
40. read(fp[0],&kq,sizeof(kq));
41. *//ghi ra file*
42. FILE \*fp;
43. fp=fopen("Result.txt","w+");
44. fprintf(fp,"%s %s %s =%d\n",argv[1],argv[3],argv[2],kq);
45. fclose(fp);
46. }
47. }
48. else{
49. printf("Pipe failed\n");
50. return -2;
51. }
52. return 0;
53. }

**2A.2 Kết quả chạy chương trình:**

****

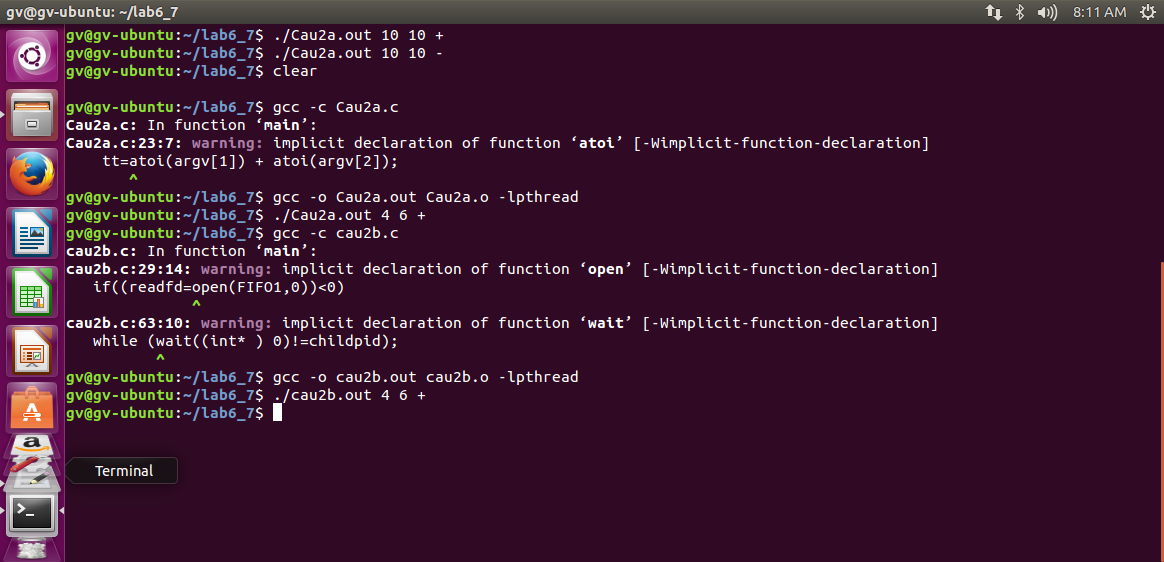
****

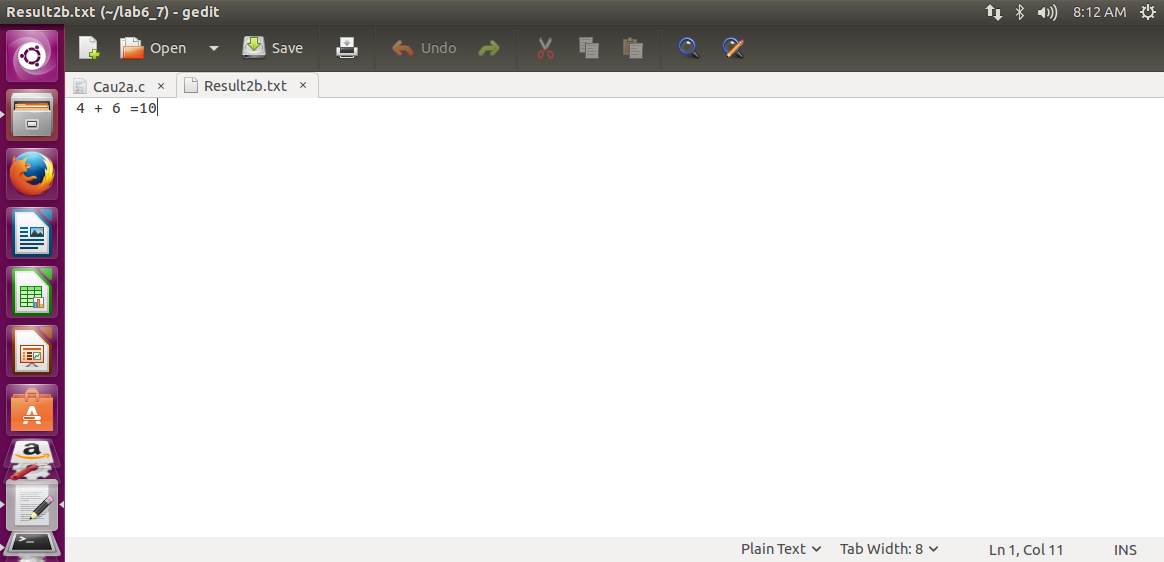
**Bài 2B.** Giải lại vấn đề 2A với đường ống có tên (Named Pipe).

**1.Phần source code:**

1. #include<stdio.h>
2. #include<stdlib.h>
3. #include<unistd.h>
4. #include<string.h>
5. #include<sys/types.h>
6. #include<sys/stat.h>
7. #include<sys/errno.h>
8. #define FIFO1 "/tmp/ff.1"
9. #define FIFO2 "/tmp/ff.2"
10. #define PM 0666
11. extern int errno;
12. #define PIPE\_BUF 4096
13. int main( int argc, char\* argv[]){
14. int kq,tt;
15. int childpid, readfd, writefd;
16. if((mknod(FIFO1,S\_IFIFO | PM,0)<0)&& (errno!=EEXIST)){
17. printf(" Fail to create FIFO 1.Aborted. \n");
18. return -1;
19. }
20. if((mknod(FIFO2,S\_IFIFO | PM,0)<0)&& (errno!=EEXIST)){
21. unlink(FIFO1);
22. printf(" Fail to create FIFO 2.Aborted. \n");
23. return -1;
24. }
25. childpid=fork();
26. if (childpid ==0){
27. if((readfd=open(FIFO1,0))<0)
28. perror(" child cannot open readFIFO \n");
29. if((writefd=open(FIFO2,1))<0)
30. perror(" child cannot open writeFIFO \n");
31. char pt=\*argv[3];
32. if(pt == '+'){
33. tt=atoi(argv[1]) + atoi(argv[2]);
34. }
35. else if(pt == '-'){
36. tt=atoi(argv[1]) - atoi(argv[2]);
37. }
38. else if(pt == '\*'){
39. tt=atoi(argv[1]) \* atoi(argv[2]);
40. }
41. else if(pt == '/'){
42. tt=atoi(argv[1]) / atoi(argv[2]);
43. }
44. else{
45. tt=0;
46. }
47. write(writefd,&tt,sizeof(tt));
48. close(writefd);
49. close(readfd);
50. return 1;
51. }
52. else if(childpid >0){
53. if((writefd=open(FIFO1,1))<0)
54. perror(" Parent cannot open writeFIFO \n");
55. if((readfd=open(FIFO2,0))<0)
56. perror(" Parent cannot open readFIFO \n");
57. read(readfd,&kq,PIPE\_BUF);
58. FILE \*fp;
59. fp=fopen("Result2b.txt","w+");
60. fprintf(fp," %s %s %s =%d\n",argv[1],argv[3],argv[2],kq);
61. while (wait((int\* ) 0)!=childpid);
62. close(writefd);
63. close(readfd);
64. if(unlink(FIFO1)<0)
65. perror("cannot remove FIFO1 \n");
66. if(unlink(FIFO2)<0)
67. perror("cannot remove FIFO2 \n");
68. return 1;
69. }
70. else{
71. printf("Fork failed \n ");
72. return -1;
73. }return 0;
74. }

**2B.2 Kết quả chạy chương trình**





**KẾT LUẬN**

Qua lab6\_7 em hiểu được cách sử dụng unnamed pipe và named pipe, cách chia sẻ vùng nhớ và hàng đợi message.