LAB 5.1

**Bài 1**

**UnName**

1. *//Bai1 UnName*
2. #include <stdio.h>
3. #include <unistd.h>
4. #include <string.h>
6. int main(int argc, char \*\*argv)
7. {
8. int fp1[2];
10. int pid;
11. *//printf("%d",argc);*
12. if (argc < 2)
13. {
14. [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("Doi so thieu.**\n**");
15. return -1;
16. }
17. if (pipe(fp1) == 0)
18. {
19. pid = fork();
21. if (pid < 0)
22. {
23. [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("Fork failed**\n**");
24. return -1;
25. }
26. else if (pid == 0)
27. {
28. close(fp1[1]);
29. char buffer[256];
30. while (read(fp1[0], &buffer, sizeof(buffer))>0)
31. {
32. [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("%s**\n**",buffer);
33. }
34. close(fp1[0]);
35. }
36. else
37. {
38. close(fp1[0]);
39. *//o doc 1 viet*
40. int i;
41. for (i = 1; i <argc; i++){
42. write(fp1[1],&argv[i],sizeof(&argv[i]));
43. }
44. *//wait(NULL);*
45. close(fp1[1]);
46. }
47. }
48. else
49. {
50. [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("Pipe failed**\n**");
51. return -2;
52. }
53. }

**Name**

1. *//Name*
2. #include <stdio.h>
3. #include <stdlib.h>
4. #include <unistd.h>
5. #include <string.h>
6. #include <sys/types.h>
7. #include <sys/stat.h>
8. #include <sys/errno.h>
9. #define FIFO1 "/tmp/ff.1"
10. #define FIFO2 "/tmp/ff.2"
11. #define PM 0666
12. **extern** int errno;
13. #define PIPE\_BUF 4096
14. int main(int argc, char \*argv[])
15. {
16. char s1[PIPE\_BUF], s2[PIPE\_BUF];
17. int childpid, readfd, writefd;
18. if ((mknod(FIFO1, S\_IFIFO | PM, 0) < 0) && (errno != EEXIST))
19. {
20. [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("Fail to create FIFO 1. Aborted.**\n**");
21. return -1;
22. }
23. if ((mknod(FIFO2, S\_IFIFO | PM, 0) < 0) && (errno != EEXIST))
24. {
25. unlink(FIFO1);
26. [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("Fail to create FIFO 2. Aborted.**\n**");
27. return -1;
28. }
29. childpid = fork();
30. if (childpid == 0)
31. { *// child*
32. if ((readfd = open(FIFO1, 0)) < 0)
33. [perror](http://www.opengroup.org/onlinepubs/009695399/functions/perror.html)("Child cannot open readFIFO.**\n**");
34. while(read(readfd, s2, PIPE\_BUF)){
35. [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("%s**\n**", s2);
36. }
37. close(readfd);
38. return 1;
39. }
40. else if (childpid > 0)
41. { *// parent*
42. if ((writefd = open(FIFO1, 1)) < 0)
43. [perror](http://www.opengroup.org/onlinepubs/009695399/functions/perror.html)("Parent cannot open writeFIFO.**\n**");
44. int i;
45. for (i = 1; i < argc; i++)
46. {
47. [gets](http://www.opengroup.org/onlinepubs/009695399/functions/gets.html)(s1);
48. write(writefd, s1, PIPE\_BUF);
49. }
50. while (wait((int \*)0) != childpid);
51. close(writefd);
52. if (unlink(FIFO1) < 0)
53. [perror](http://www.opengroup.org/onlinepubs/009695399/functions/perror.html)("Cannot remove FIFO1.**\n**");
54. return 1;
55. }
56. else
57. {
58. [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("Fork failed**\n**");
59. return -1;
60. }
61. **}**

**Bài 2**

**UnName**

1. *//UnName*
2. #include <stdio.h>
3. #include <unistd.h>
4. #include <string.h>
6. int main(int argc, char \*argv[])
7. {
8. int fp1[2], fp2[2];
9. int pid;
10. if (argc < 2)
11. {
12. [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("Doi so thieu.**\n**");
13. return -1;
14. }
15. if (pipe(fp1) == 0)
16. {
17. pid = fork();
19. if (pid < 0)
20. {
21. [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("Fork failed**\n**");
22. return -1;
23. }
24. else if (pid == 0)
25. {
26. int buffer;
27. close(fp1[1]);
28. read(fp1[0], &buffer, sizeof(buffer));
29. int cnt = 1, i;
30. for (i = 1; i <= buffer; i++)
31. {
32. cnt \*= i;
33. }
34. [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("%d!=%d**\n**", buffer, cnt);
35. close(fp1[0]);
36. }
37. else
38. {
39. close(fp1[0]);
40. *// o doc 1 viet*
41. int tmp = [atoi](http://www.opengroup.org/onlinepubs/009695399/functions/atoi.html)(argv[1]);
42. write(fp1[1], &tmp, sizeof(tmp));
43. close(fp1[1]);
44. }
45. }
46. else
47. {
48. [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("Pipe failed**\n**");
49. return -2;
50. }
51. }

**Name**

1. *//Name*
2. #include <stdio.h>
3. #include <stdlib.h>
4. #include <unistd.h>
5. #include <string.h>
6. #include <sys/types.h>
7. #include <sys/stat.h>
8. #include <sys/errno.h>
9. #define FIFO1 "/tmp/ff.1"
10. #define FIFO2 "/tmp/ff.2"
11. #define PM 0666
12. **extern** int errno;
13. #define PIPE\_BUF 4096
14. int main(int argc, char \*argv[])
15. {
16. char s1[PIPE\_BUF], s2[PIPE\_BUF];
17. int childpid, readfd, writefd;
18. if ((mknod(FIFO1, S\_IFIFO | PM, 0) < 0) && (errno != EEXIST))
19. {
20. [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("Fail to create FIFO 1. Aborted.**\n**");
21. return -1;
22. }
23. if ((mknod(FIFO2, S\_IFIFO | PM, 0) < 0) && (errno != EEXIST))
24. {
25. unlink(FIFO1);
26. [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("Fail to create FIFO 2. Aborted.**\n**");
27. return -1;
28. }
29. childpid = fork();
30. if (childpid == 0)
31. { *// child*
32. if ((readfd = open(FIFO1, 0)) < 0)
33. [perror](http://www.opengroup.org/onlinepubs/009695399/functions/perror.html)("Child cannot open readFIFO.**\n**");
34. [fflush](http://www.opengroup.org/onlinepubs/009695399/functions/fflush.html)(stdin);
35. read(readfd, s2, PIPE\_BUF);
36. int cnt = 1;
37. int i;
38. for (i = 1; i <= [atoi](http://www.opengroup.org/onlinepubs/009695399/functions/atoi.html)(s2); i++)
39. {
40. cnt \*= i;
41. }
42. [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("%d!=%d**\n**", [atoi](http://www.opengroup.org/onlinepubs/009695399/functions/atoi.html)(s2), cnt);
43. close(readfd);
44. return 1;
45. }
46. else if (childpid > 0)
47. { *// parent*
48. if ((writefd = open(FIFO1, 1)) < 0)
49. [perror](http://www.opengroup.org/onlinepubs/009695399/functions/perror.html)("Parent cannot open writeFIFO.**\n**");
50. [fflush](http://www.opengroup.org/onlinepubs/009695399/functions/fflush.html)(stdin);
51. [scanf](http://www.opengroup.org/onlinepubs/009695399/functions/scanf.html)("%s",s1);
52. write(writefd, s1,[strlen](http://www.opengroup.org/onlinepubs/009695399/functions/strlen.html)(s1));
53. while (wait((int \*)0) != childpid)
54. ;
55. close(writefd);
56. if (unlink(FIFO1) < 0)
57. [perror](http://www.opengroup.org/onlinepubs/009695399/functions/perror.html)("Cannot remove FIFO1.**\n**");
58. return 1;
59. }
60. else
61. {
62. [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("Fork failed**\n**");
63. return -1;
64. }
65. }

**Bài 3**

**Name**

1. *//Bai3 UnName*
2. #include <stdio.h>
3. #include <unistd.h>
4. #include <string.h>
6. int main(int argc, char \*argv[])
7. {
8. int fp1[2], fp2[2];
9. int pid;
10. char arr[argc + 1];
11. if (argc < 2)
12. {
13. [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("Doi so thieu.**\n**");
14. return -1;
15. }
16. if (pipe(fp1) == 0)
17. {
18. pid = fork();
20. if (pid < 0)
21. {
22. [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("Fork failed**\n**");
23. return -1;
24. }
25. else if (pid == 0)
26. {
27. char buffer[100];
28. close(fp1[1]);
29. char arr[100];
30. int cnt = 0;
32. [fflush](http://www.opengroup.org/onlinepubs/009695399/functions/fflush.html)(stdin);
33. *//read(fp1[0], &buffer, sizeof(buffer)) ;*
34. *//printf("%s",buffer);*
35. *//scanf("%s%s%s",arr[0],arr[1]),arr[2]);*
36. while (read(fp1[0], &buffer, sizeof(buffer))!= 0)
37. {
38. [strcpy](http://www.opengroup.org/onlinepubs/009695399/functions/strcpy.html)(arr[cnt],buffer);
39. cnt++;
40. *//printf("%s",buffer);*
41. }
42. */\**
43. *switch(arr[2]){*
44. *case '+':*
45. *printf("%d+%d=%d\n",atoi(arr[0]),atoi(arr[1]),atoi(arr[0]) + atoi(arr[1]));*
46. *break;*
47. *case '-':*
48. *printf("%d-%d=%d\n",atoi(arr[0]),atoi(arr[1]),atoi(arr[0]) - atoi(arr[1]));*
49. *break;*
50. *case '\*':*
51. *printf("%d\*%d=%d\n",atoi(arr[0]),atoi(arr[1]),atoi(arr[0]) \* atoi(arr[1]));*
52. *break;*
53. *case '/':*
54. *if(atoi(arr[1])==0){*
55. *printf("Khong chia duoc cho 0\n");*
56. *}else{*
57. *printf("%d/%d=%d\n",atoi(arr[0]),atoi(arr[1]),atoi(arr[0]) / atoi(arr[1]));*
58. *}*
59. *break;*
60. *default:*
61. *printf("Khong co toan tu\n");*
62. *break;*
63. *}*
64. *\*/*
65. close(fp1[0]);
66. }
67. else
68. {
69. close(fp1[0]);
70. int i;
71. [fflush](http://www.opengroup.org/onlinepubs/009695399/functions/fflush.html)(stdin);
72. for (i = 1; i < argc; i++)
73. {
74. write(fp1[1], argv[i], [strlen](http://www.opengroup.org/onlinepubs/009695399/functions/strlen.html)(argv[i]));
75. }
76. close(fp1[1]);
77. }
78. }
79. else
80. {
81. [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("Pipe failed**\n**");
82. return -2;
83. }
84. }

**Name**

1. *//Name*
2. #include <stdio.h>
3. #include <stdlib.h>
4. #include <unistd.h>
5. #include <string.h>
6. #include <sys/types.h>
7. #include <sys/stat.h>
8. #include <sys/errno.h>
9. #define FIFO1 "/tmp/ff.1"
10. #define FIFO2 "/tmp/ff.2"
11. #define PM 0666
12. **extern** int errno;
13. #define PIPE\_BUF 4096
14. int main(int argc, char \*argv[])
15. {
16. char s1[PIPE\_BUF], s2[PIPE\_BUF];
17. int childpid, readfd, writefd;
18. if ((mknod(FIFO1, S\_IFIFO | PM, 0) < 0) && (errno != EEXIST))
19. {
20. [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("Fail to create FIFO 1. Aborted.**\n**");
21. return -1;
22. }
23. if ((mknod(FIFO2, S\_IFIFO | PM, 0) < 0) && (errno != EEXIST))
24. {
25. unlink(FIFO1);
26. [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("Fail to create FIFO 2. Aborted.**\n**");
27. return -1;
28. }
29. childpid = fork();
30. if (childpid == 0)
31. { *// child*
32. if ((readfd = open(FIFO1, 0)) < 0)
33. [perror](http://www.opengroup.org/onlinepubs/009695399/functions/perror.html)("Child cannot open readFIFO.**\n**");
34. char arr[4];
35. int cnt=0;
36. [fflush](http://www.opengroup.org/onlinepubs/009695399/functions/fflush.html)(stdin);
37. while (read(readfd, s1, PIPE\_BUF) > 0){
38. arr[cnt++] = s1[0];
39. }
40. [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("%s",s1[0]);
41. close(readfd);
42. return 1;
43. }
44. else if (childpid > 0)
45. { *// parent*
46. if ((writefd = open(FIFO1, 1)) < 0)
47. [perror](http://www.opengroup.org/onlinepubs/009695399/functions/perror.html)("Parent cannot open writeFIFO.**\n**");
48. int i;
49. for (i = 0; i < argc; i++)
50. {
51. [fflush](http://www.opengroup.org/onlinepubs/009695399/functions/fflush.html)(stdin);
52. [scanf](http://www.opengroup.org/onlinepubs/009695399/functions/scanf.html)("%s",s1);
53. write(writefd, s1, PIPE\_BUF);
54. }
55. while (wait((int \*)0) != childpid)
56. ;
57. close(writefd);
58. if (unlink(FIFO1) < 0)
59. [perror](http://www.opengroup.org/onlinepubs/009695399/functions/perror.html)("Cannot remove FIFO1.**\n**");
60. return 1;
61. }
62. else
63. {
64. [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("Fork failed**\n**");
65. return -1;
66. }
67. }