Brian Duenas

CSE 460

Lab 4

20 points Total

2. **Process Pipes**

Q: What do you see when you execute "pipe1" ? Why?

A: We see the "*ps –auxw*" execute and print on screen. The program holds the command in *buffer* then it is printed on screen.

Q: Modify the program pipe1.cpp to pipe1a.cpp so that it accepts a command (e.g. "ls -l") from the keyboard. For example, when you execute "./pipe1a ps -auxw", it should give you the same output as pipe1.cpp.

A:

```
//pipe1a.cpp
#include <unistd.h>
#include <stdlib.h>
#include <string.h>
#include <stdio.h>
#include <iostream>

using namespace std;

int main(int argc, char *argv[]) {
{
        char input[50];

        strcpy(input, argv[1]);  //get first instance of command

        for (int i = 2; i <= (argc - 1); i++) {  //get command params
               strcat(input, " ");
               strcat(input, argv[i]);
        }

        FILE *fpi;     //for reading a pipe

        char buffer[BUFSIZ + 1]; //BUFSIZ defined in <stdio.h>

        int chars_read;
        memset(buffer, 0, sizeof(buffer));  //clear buffer
        fpi = popen(input, "r");     //pipe to command "ps -auxw"
        if (fpi != NULL) {
               //read data from pipe into buffer
               chars_read = fread(buffer, sizeof(char), BUFSIZ, fpi);
               if (chars_read > 0)
                      cout << "Output from pipe: " << buffer << endl;
               pclose(fpi);                  //close the pipe
               return 0;
        }

        return 1;
}
```

Example:

```
[user@csusb.edu@jb359-3 lab4]$ pipe1a ls -l
Output from pipe : total 57
- rwxr - xr - x 1 005029683@csusb.edu domain users@csusb.edu 9088 Apr 25 11:18 p
ipe1
- rwxr - xr - x 1 005029683@csusb.edu domain users@csusb.edu 9192 Apr 25 13:39 p
ipe1a
- rw - r--r-- 1 005029683@csusb.edu domain users@csusb.edu  792 Apr 25 13:39 p
ipe1a.cpp
- rw - r--r-- 1 005029683@csusb.edu domain users@csusb.edu  657 Apr 25 11:17 p
ipe1.cpp
```

Q: What do you see when you execute "pipe2" ? Why?

Output:

```
[005029683@csusb.edu@jb359-3 lab4]$ pipe2
0000000   A   r   n   o   d       s   a   i   d,  '   I   f
0000020   I       a   m       e   l   e   c   t   e   d,  .   .
0000040   '   ,       a   n   d       t   h   e       f   a   i   r   y
0000060       t   a   l   e       b   e   g   i   n   s   \n
0000075
```

A: The string from the program is printed as the "*od –c*" command is executing.

Q: Modify the program so that it prints out the first three words of the sentence in reverse by making use of awk (see lab 2) (i.e. 'If said, Arnod....).

A:

```
//pipe2a.cpp
#include <unistd.h>
#include <stdlib.h>
#include <string.h>
#include <stdio.h>
#include <iostream>

using namespace std;

int main()
{
        FILE *fpo;                                  //for writing to a pipe
        char buffer[BUFSIZ + 1];                      //BUFSIZ defined in <stdio.h>

        sprintf(buffer, "Arnod said, 'If I am elected, ..', and the fairy tale begins");

        fpo = popen("awk '{t=$1;$1=$3;$3=t} 1' ", "w");    //pipe to command "od -c"

        if (fpo != NULL) {
                //send data from buffer to pipe
                fwrite(buffer, sizeof(char), strlen(buffer), fpo);
                pclose(fpo);                        //close the pipe
                return 0;
        }
        return 1;
}
```

Output:

```
[user@csusb.edu@jb359-1 lab4]$ pipe2a
'If said, Arnod I am elected, ..', and the fairy tale begins
[user@csusb.edu@jb359-1 lab4]$
```

3. The pipe Call

Q: What do you see when you execute "pipe3" ? Why?

Output:

```
[005029683@csusb.edu@jb359-3 lab4]$ pipe3
Sent 5 bytes to pipe.
Read 5 from pipe : CSUSB
```

A: You made an array of size two and sent "*CSUSB*" to array from pipe. Then the pipe is read using the unused array space using pipe.

4. Parent and Child Processes

Q: Modify pipe4.cpp so that it accepts a message from the keyboard and sends it to pipe5.

A:

```
//pipe4.cpp  (data producer)
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>

int main(int argc, char *argv[])
{
        char user[100];
        strcpy(user, argv[1]);

        for (int i = 2; i <= (argc - 1); i++) {
                strcat(user, " ");
                strcat(user, argv[i]);
        }

        int data_processed;
        int file_pipes[2];
        const char some_data[] = "123";
        char buffer[BUFSIZ + 1];
        pid_t fork_result;

        memset(buffer, '\0', sizeof(buffer));

        if (pipe(file_pipes) == 0) {   //creates pipe
                fork_result = fork();
                if (fork_result == (pid_t)-1) {  //fork fails
                        fprintf(stderr, "Fork failure");
                        exit(EXIT_FAILURE);
                }

                if (fork_result == 0) {     //child
                        sprintf(buffer, "%d", file_pipes[0]);
                        (void)execl("pipe5", "pipe5", buffer, (char *)0);
                        exit(EXIT_FAILURE);
                }
                else {                      //parent
                        data_processed = write(file_pipes[1], user,
                                strlen(user));
                        printf("%d - wrote %d bytes\n", getpid(), data_processed);
                }
        }
        exit(EXIT_SUCCESS);
}
```

Output:

```
[user@csusb.edu@jb359-3 lab4]$ pipe4 this sentence is in pipe
31473 - wrote 24 bytes
31474 - read 24 bytes: this sentence is in pipe
```

5. Special Pipes

Q: modify the scripts so that received characters are converted to lower case rather than upper case.

A: Changed *server.cpp* so that it is not sending upper case characters by removing *toupper()*.

```
tmp_char_ptr = my_data.some_data;
while (*tmp_char_ptr) {
        *tmp_char_ptr = tolower(*tmp_char_ptr);
        tmp_char_ptr++;
}
sprintf(client_fifo, CLIENT_FIFO_NAME, my_data.client_pid);
```

Output:

```
22982 sent Hello from 22982, received: hello from 22982
22982 sent Hello from 22982, received : hello from 22982
22982 sent Hello from 22982, received : hello from 22982
22982 sent Hello from 22982, received : hello from 22982
22982 sent Hello from 22982, received : hello from 22982
```

6. Study of XV6

```
#include "types.h"
#include "stat.h"
#include "user.h"
#include "fcntl.h"

char buf[512];
int
main(int argc, char *argv[])
{
        int fd0, fd1, n;

        if (argc <= 2) {
                printf(1, "need 2 arguments!\n");
                exit();
        }
        for (int i = 2; i <= argc; i++) {
                if ((fd0 = open(argv[1], O_RDONLY)) < 0) {
                        printf(1, "CP: cannot open %s\n", argv[1]);
                        exit();
                }
                if ((fd1 = open(argv[i], O_CREATE | O_RDWR)) < 0) {
                        printf(1, "CP: cannot open %s\n", argv[i]);
                        exit();
                }
                while ((n = read(fd0, buf, sizeof(buf))) > 0) {
                        write(fd1, buf, n);
                }
                close(fd0);
                close(fd1);
        }
        exit();
}
```

Output:

```
.              1 1 512
..             1 1 512
README         2 2 2290
cat            2 3 13680
echo           2 4 12688
forktest       2 5 8124
grep           2 6 15556
init           2 7 13276
kill           2 8 12740
ln             2 9 12644
ls             2 10 14828
mkdir          2 11 12820
rm             2 12 12804
sh             2 13 23288
stressfs       2 14 13468
usertests      2 15 56404
wc             2 16 14220
cp             2 17 13424
zombie         2 18 12468
console        3 19 0
$ cp README file1 file2
ls
$ .              1 1 512
..             1 1 512
README         2 2 2290
cat            2 3 13680
echo           2 4 12688
forktest       2 5 8124
grep           2 6 15556
init           2 7 13276
kill           2 8 12740
ln             2 9 12644
ls             2 10 14828
mkdir          2 11 12820
rm             2 12 12804
sh             2 13 23288
stressfs       2 14 13468
usertests      2 15 56404
wc             2 16 14220
cp             2 17 13424
zombie         2 18 12468
console        3 19 0
file1          2 20 2290
file2          2 21 2290
$
```