COMP 7500 Advanced Operating Systems

Project 2: pWordCount: A Pipe-based WordCount Tool

Frequently Asked Questions

VERSION 2.9
REVISED: February 5, 2025

1. How to count words in a text?

Answer: You may count the number of spaces. Please check the sample code here:
http://www.sanfoundry.com/c-program-count-words-in-sentence/

Please also refer to the following discussions documented on `stackoverflow`.
https://stackoverflow.com/questions/25746334/count-words-in-a-user-input-string-in-c

2. Should we consider numbers while counting words?

Answer: Yes, you may consider numbers as a word, because we are counting the number of spaces. For example,

"There were 6 people": 4 words
"It was a gr8 day": 5 words.

3. How can I access a remote Linux machine in our college's computer laboratories?

Answer: If you plan to remotely access Linux machines in our college's computer lab, please refer to the instructions here: Programming Environment Setup.pdf.
Module "Lunix Programming" on Canvas contains all the information about programming in the Linux environment.

4. Do we need to account for whitespace such as tab and newline, etc?

Answer: The project specification doesn't have any requirements on (1) special characters (e.g., numbers and punctuation) and (2) upper/lower cases. You may choose to (1) ignore these issues or (2) put an extra effort to handle these cases. I believe your programming skills will be enhanced if you take option (2).

5. I have a Mac. Can I use LLDB rather than GDB as a debugger?

Answer: Yes, you can use LLDB to perform the debugging task on your local Mac.

6. My child process won't run because the parent process ends too fast.

Answer: You should make the parent process waits for all the child processes to finish. Please follow the sample code here to solve this problem: https://stackoverflow.com/questions/19461744/how-to-make-parent-wait-for-all-child-processes-to-finish

7. **C test framework.** Is there a specific unit testing framework that we should be using? (Spring'20)

Answer: I recommend you to use cunit. See sample code and sample usages here: http://cunit.sourceforge.net/. You may, of course, choose to use any unit testing framework such as the GoogleTest framework (Reference: Unit Testing C code with the GoogleTest framework).

8. **Integer in Pipes.** Can we pass integers between the parent and child processes? (Spring'21)
Answer: You can follow the sample code below to send and receive integers between two processes.

```
int  val = 100;
/* Send integer val */
write(fd[1], &val, sizeof(val));

/* Receive integer val */
read(fd[0], &val, sizeof(val));
```

9. **Comment Block.** Can you provide a sample comment block for our source code files? (Spring'21)
Answer: The sample comment block is given blow.

```
/*
 * COMP 7500/06: Project 5 Scheduling
 * Xiao Qin
 * Auburn University
 *
 * This source code shows how to pass commandline arguments to your progrm
 *
 * How to compile?
 * $gcc command.c -o command
 *
 * How to run?
 * Case 1: no argument. Sample usage is printed
 * $./command
 * Usage: command file_name [FCFS|RR|SRTF] [time_quantum]
 *
 * Case 2: 2 arguments.
 * $./command file1 FCFS
 * The file name is: file1
 * The chosen policy is: FCFS
 *
 * Case 3: 2 arguments.
 * $./command file2 RR
 * The file name is file2
 * The chosen policy is RR
 * Please enter time_quantum for the RR policy!
 */
```

10. **GitHub and VS Code Integration.** How can I use GitHub and VS code under the Docker centos? (Contributed by Libo Sun and Alex A Knipper, Spring'22)

Answer: From the YouTube video, Libo Sun provided a solution of using VS Code as the editor and transferring sources codes between host computer and the docker centos via GitHub.

YouTube Video: [Git and VS Code between host and docker centos 7 - YouTube]

Note that:

- This solution might not be the best one (less connection issues)
- The example repository is private and you may not access it
- The token example has been removed after making the video.

Comment from Alex A Knipper:

- If you're using Docker and VSCode, you can also install the Docker extension to just remotely connect to the container and edit the code directly.
- Another option, if you are comfortable reconfiguring the container, is to use a bind mount to directly map a directory in the host OS to the container (this is the approach I typically like using)

If you would like to learn how to use GitHub with VS Code, please watch this YouTube tutorial here: https://www.youtube.com/watch?v=3Tn58KQvWtU

11. **Segmentation Fault Error.** I had a "segmentation fault(core dumped)" error when working on this project. How can I fix it? I am using a Mac and running centOS on VMware. (Contributed by Joshua Charles Boyd, Spring'23)

    Answer: Common reasons for segmentation fault include accessing an array out of bounds and improper formatting specifiers in printf and scanf statements.

    This is where using gdb really comes in handy. First make sure you are compiling your program with the -g flag for debugging

    ```
    gcc -g -o pwordcount pwordcount.c
    ```

    next start gdb with your program

    ```
    gdb ./pwordcount
    ```

    then using gdb you can set a breakpoint where you think the segmentation fault might be occurring. For example if you thought it was around line 30 in your code you would do this.

    ```
    (gdb) b 30
    ```

    afterwards you run your program with whatever text file. The program should stop at the breakpoint and look something like this

    ```
    (gdb) run testFile.txt
    ```

    Starting program: `/home/jboyd/AdvOpProject2/pwordcount testFile.txt`
    warning: Error disabling address space randomization: Operation not permitted
    [Detaching after fork from child process 45]
    Process 1 is reading file "testFile.txt" now...

    ```
    Breakpoint 1, loadFile (file_name=0xffffd2b4a7d0 "testFile.txt",
    file_size=76, write_msg_buffer=0xffffd2b4a7b8) at
    pwordcount.c:30
    warning: Source file is more recent than executable.
    30      *write_msg_buffer = calloc(1, (file_size+1) *
    sizeof(char));
    (gdb)
    ```

what you see toward the end is the line of code I have at line 30. From here you can step through your code's execution line by line using the 'next' (or 'n') command until you hit the actual segmentation fault.

```
30      *write_msg_buffer = calloc(1, (file_size+1) *
sizeof(char));
(gdb) n
31      fread(*write_msg_buffer, 1, file_size, fp);
(gdb) n
32      fclose(fp);
(gdb) n
33      return 0;
(gdb) n
34 }
(gdb)
```

basically mess around with your program in this way until you find the exact line where the segmentation fault is happening.
I guess if you had no idea where the segmentation fault might be happening to begin with, you could just set a breakpoint at line 1 and step through your entire program.

12. **WordCount Design Question.** When the parent process delivers data to its child process we could do it in two ways. Either we can send the whole file content in one call or we can split the file into chucks and send it via multiple calls. There is definitely some advantages and disadvantage in both approaches. Just wanted to know what is expected? (Spring'24)

    Answer: You are right: there are two main approaches to delivering file data to a child process -- sending the entire content in one call or splitting it into chunks and sending multiple calls. Both methods have their own advantages and disadvantages, depending on the specific context.
    For example, sending the entire file in a single call has the following pros and cons.

    Pros: (1) it is easier to implement. (2) for small files, it is a fast approach.
    Cons: (1) Requires enough memory to hold the entire file: it will an issue for very large files. (2) Again, for large files, the parent process is blocked until the child finishes reading the entire file -- response time is enlarged. (3) If transfer fails, the entire file needs to be sent again.

    Following the above pros and cons, you can analyze the advantages and disadvantages of splitting a file into small chunks.

    Ideally, you should implement approach 1 for small files -- and develop approach 2 for large files. If you would like to quickly carryout project 2, approach one is a better choice. In case that you have a desire to strengthen your programming skills, you should opt for a hybrid approach. Nonetheless, it is your decision to elect for approach 1, approach 2, or a hybrid approach.

13. **Check File Format.** How can I test if an input file is a text file or a binary file?

    Answer: In C programming, there's no built-in standard function to directly determine whether a file is a text or binary file. However, you can employ some heuristics to make an educated guess based on the content of the file. One common idea is to analyze the content of the file and check for certain characters or patterns that are typical in text files but less likely to occur in

binary files. For instance, you may check for the presence of null bytes (0x00), control characters, or a high density of non-printable ASCII characters.

Please find the sample code – is_text_file.c – on Canvas, and here is the path to locate the sample source code file: Canvas->Files->Projects->Project 2-A Pipe-based WordCount Tool->sample_code->is_text_file.c

14. **Unit Test Results.** Should we show the cunit unit test cases runs or is it good enough to have that part in the source code but keep it commented (the unit test cases in reality is run during the test phase of a program and once its production ready its comment out or ignored with a compile time option)?

    Answer: It is optional to show cunit test results in your report if you choose to do so. Nonetheless, undertaking unit testing results is a good approach to improving your programming skills. You may define a UNIT_TEST macro so that you and the GTA can easily switch between the UNIT_TEST version and the product version.

    The sample code demonstrating how to use CUNIT can be found from "Files" or "Modules" on Canvas below:
    (1) Canvas->Files->Projects->CUNIT
    (2) Canvas->Modules->How to use CUnit to conduct unit testing?

15. **Single Line.** Can I assume that an input text file has all text on a single line?

    Answer: No, you can't assume that the input file has all text on a single line. Your code should consider '\n' and '\r', which signify new lines.

16. **Creating Two Pipes.** To create a second pipe, does it had to be done by the child process to connect to the parent? (Spring'25)

    Answer: The two pipes can be created in the parent process. Please follow the sample code below to create two pipes.

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

int main() {
    int pipe1[2], pipe2[2]; // pipe1: Parent -> Child, pipe2: Child ->
Parent
    pid_t pid;
    char buffer[100];

    if (pipe(pipe1) == -1 || pipe(pipe2) == -1) {
        perror("pipe");
        exit(EXIT_FAILURE);
    }
```

```
    pid = fork();

    if (pid == -1) {
        perror("fork");
        exit(EXIT_FAILURE);
    }

    if (pid == 0) {  // Child process
        close(pipe1[1]); // Close write end of parent -> child pipe
        close(pipe2[0]); // Close read end of child -> parent pipe

        read(pipe1[0], buffer, sizeof(buffer)); // Read from parent
        printf("Child received: %s\n", buffer);

        write(pipe2[1], "Message from child", 18); // Send to parent
        close(pipe1[0]);
        close(pipe2[1]);
    } else {  // Parent process
        close(pipe1[0]); // Close read end of parent -> child pipe
        close(pipe2[1]); // Close write end of child -> parent pipe

        write(pipe1[1], "Hello from parent", 17); // Send to child
        read(pipe2[0], buffer, sizeof(buffer)); // Read from child
        printf("Parent received: %s\n", buffer);

        close(pipe1[1]);
        close(pipe2[0]);
    }

    return 0;
}
```

17. **Differences between Small and Large Files.** What would be an example of a large file in comparison to a small one? (Spring'25)

    Answer: Typically, a small file might be a text file or a configuration file that is a few KB in size. For example, a file containing a few hundred lines of code or a simple text document. In contrast, a large file could be a log file or a dataset file that is several GBs in size. For example, a large log file with millions of records.

18. **CUnit Installation.** I am trying to install CUnit on my Linux-Ubuntu terminal in my Windows PC, but I can't run ./configure. How to solve this problem. (Spring'25)

    Answer: You can install CUnit using the following command:

    ```
    sudo apt update
    sudo apt install libcunit1-dev
    ```

    If the above solution doesn't work, please try the backup plan below

    ```
    #Install build dependencies
    sudo apt update
    sudo apt install build-essential autoconf automake libtool

    #Download CUnit Source
    ```

```
wget
https://sourceforge.net/projects/cunit/files/latest/downloa
d -O CUnit.tar.bz2
tar -xvf CUnit.tar.bz2
cd CUnit-*

#install CUnit
sudo apt update
sudo apt install libcunit1-dev

#Run Autotools go create configure
autoreconf -i
./configure

#Then,
make
sudo make install
```