# Pipe-Sort (due 10/27)

## Learning Objectives

Upon completion of this assignment, you should be able:

1. Work in a multiprocess environment
2. Combine newly written processes with existing system utility processes
3. Gain facility with interprocess communication

New mechanisms you will see and use include:

- system calls: fork, exec, wait, pipe, dup2
- /usr/bin/sort, buffered I/O, String Routines

NAME

pipesort – An exercise in plumbing

SYNOPSIS

pipesort [n:s:l:]

DESCRIPTION

| | |
|---|---|
| -n count | Count is the number of sorters (default 1) |
| -s short | Words must have more than "short" letters |
| -l long | "long" is the maximum number of letters in a word.Letters beyond that are discarded |

pipesort reads text from standard input and writes the unique words to standard output sorted in alphabetic order, using "count sorters".

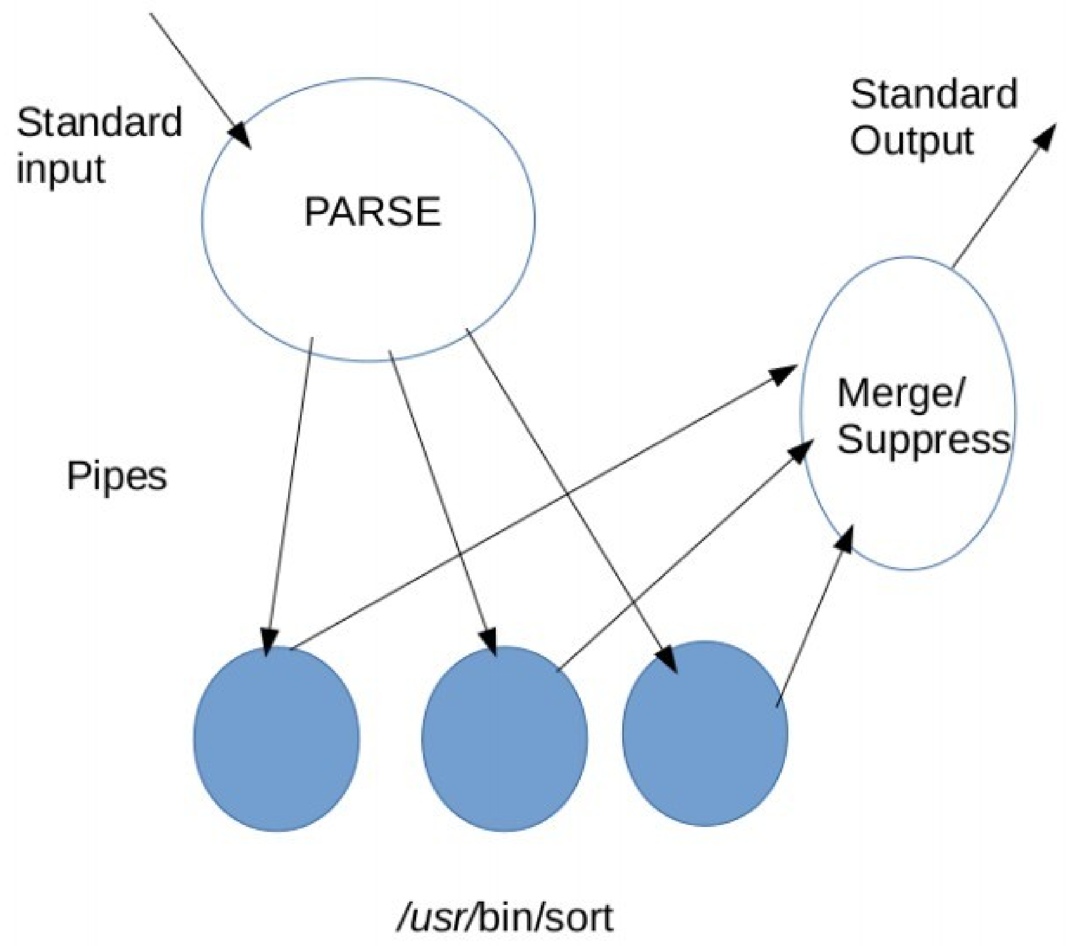
Exit Status

pipesort returns 0 on success and 1 on failure

DISCUSSION

The output should have one word per line preceded with a count of how many times it appeared in the input.

Internally, the program must be organized into 3 types of processes. A single process reads the input parsing the lines into words, a group of processes does the sorting, and a single process suppresses and counts duplicate words and writes the output. You should organize the processes so that every parent waits for its children, and if any process exits in error the main process returns the error code in its exit.



NOTES:

- You must use the system sort command (/usr/bin/sort) with no arguments to do the actual sorting,

- You must not use any disk files, or large amounts of memory. There is NO LIMIT on how large the input may be, and you should test with large input files.

- If there is more than one sorter, Parse distributes the words round robin to the sorts

- All the I/O is done using the buffered I/O library (fget, fputc, etc), you will need fdopen for attaching to the pipes.

- Words are all alphabetic and case insensitive with the parser converting all alphabetic characters to lower case. Any non-alphabetic characters delimit words and are discarded.

- The “short” and “long” parameters affect the parsing. Words with “short” characters or less are discarded, words with “long” characters or greater are truncated to “long” characters.

- The output MUST have the count left justified in a 10 character field followed immediately by the word.

Example  short=2 long=6

```
This is nonsense abc-def$@#abc
some more of this

2         abc
1         def
1         more
1         nonsen
1         some
2         this
```

HAND in pipesort.c, makefile, any other files (README) in th <yourid>/cs551/lab3 directory. Please make sure your makefile names the compiled program “pipesort”.