

Hengnan Ma, Fakharyar Khan
October 27th, 2023
Professor Hakner
Operating Systems

Three Command Pipeline

Instructions on Running the Program

To run the program, you need to compile the three programs, wordGen.c, wordSearch.c, and pager.c as executables named wordgen, wordsearch, and pager respectively. You will also need to have a file containing a list of words that will be used as a dictionary by the wordSearch program. This file should be in the same directory as the launcher program and should be named words.txt. You should then compile the launcher program. The program can take one optional argument which tells the program how many words it should generate. If no argument is generated then the program will endlessly generate words.

wordGen.c

Wordgen.c is a program that generates a specified number of random words that sends those words to standard output. These words contain 3 to nc characters (nc is hard-coded to be 10) and are composed of uppercase letters sent to the terminal, one word per line. The program takes in 1 argument from the terminal indicating the number of words to be generated. If this argument is not provided or set to 0, it generates words indefinitely. Below is an example of running this program in the terminal.

```
hengnan@DESKTOP-HEBLQU8:~/ece357/p4/wordGen$ ./wordgen 10
RWTSYXXEJFIF
BHXGTI
OJUKGU
ZPR
IQPAQNZWWHE
GHMENGMDWV
GDTO
KIZUAPWS
WPCFT
LDRP
Finished generating 10 candidate words
```

wordSearch.c

Wordsearch.c reads a dictionary of English words from a file, provided in the command line, converts them to uppercase, and stores them in memory. It then reads lines of text from standard input, checks if the words in the input match any words from the dictionary, ignoring it if it's not a match, and echos it to stdout if there is a match. The program also keeps track of the number of accepted and rejected words from the dictionary. When the program encounters a broken pipe, it will report the number of words matched before terminating.

```
hengnan@DESKTOP-HEBLQU8:~/ece357/p4/wordGen$ ./wordsearch words.txt
Accepted 416290 words, rejected 48830
hello          input
hello
word           input
word
alphabet       input
alphabet
fndsajnfdsajf input
xzvhxzuvwera  input
```

Pager.c

This program reads from standard input and prints it out line by line into standard output. After every 23 lines printed to standard output, it prompts the user to press return if they would like to see more of what's in standard input. If instead of pressing return, the user presses q or Q, the program will terminate. It should be noted that the program reads the user's inputs from the terminal (/dev/tty) instead of from standard input.

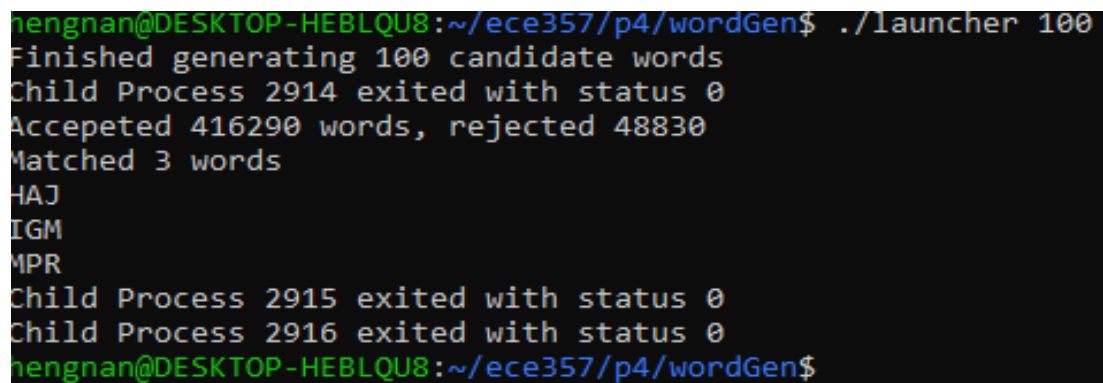
```
hengnan@DESKTOP-HEBLQU8:~/ece357/p4/wordGen$ ./pager <words.txt
a
a.
a-
AA
AAA
AAAA
AAAAAA
AAAL
AAAS
Aaberg
Aachen
AAE
AAEE
AAF
AAG
aah
aahed
aahing
aahs
AAIL
aal
Aalborg
Aalesund
---Press RETURN for more---
aalii
aaliis
aals
Aalst
Aalto
AAM
AAMSI
Aandahl
A-and-R
Aani
AAO
AAP
AAPSS
Aaqbiye
Aar
Aara
Aarau
AARC
aardvark
aardvarks
aardwolf
aardwolves
Aaren
---Press RETURN for more---
Q
hengnan@DESKTOP-HEBLQU8:~/ece357/p4/wordGen$
```

launcher.c

This program launches the above three programs as child processes and executes them in such a way that the output of wordGen is fed into the input of wordSearch whose output is in turn fed into the input of the pager program. The program can take in an optional argument that tells it how many random words it should have wordGen generate. If no arguments are given then the program will endlessly generate words. To give an overview of the process, the randomly generated words will get fed into wordSearch which will check to see if the words are in its dictionary. If a match is found then wordSearch will echo the word into the standard input of the pager program. The pager program will print out these matches 23 lines at a time.

However, since these programs are connected to each other through pipes, all reads and writes to and from the buffer are done 4096 bytes at a time. This means that because wordSearch takes much longer to search for a word than it takes for wordGen to produce a word, wordGen would have to wait for wordSearch to have processed all of the words. Likewise, wordSearch is also a slow writer in the sense that because we're producing these words randomly, it's not very likely that we will get that many matches. This means that the pager program would have to wait for the wordSearch program to find 4096 matches or to finish reading from wordGen's output.

Test Case #1: Finite Number of Generated Words



```
hengnan@DESKTOP-HEBLQU8:~/ece357/p4/wordGen$ ./launcher 100
Finished generating 100 candidate words
Child Process 2914 exited with status 0
Accepted 416290 words, rejected 48830
Matched 3 words
HAJ
IGM
MPR
Child Process 2915 exited with status 0
Child Process 2916 exited with status 0
hengnan@DESKTOP-HEBLQU8:~/ece357/p4/wordGen$
```

The figure above shows an example of the launcher program when it only generates a finite number of words. Since the number of words that wordGen generates is smaller than the size of the pipe buffer, it terminates very quickly. The wordSearch program takes a longer time since it has to search through its entire dictionary to check for a match. Since the write end of the pipe becomes closed, once it finishes processing all 100 words, it reaches the end of file and informs the user of the total number of matches found, which in this case is 3. Those 3 matched words get echoed to the pager program which in turn get echoed to standard out. Since the wordSearch program terminates after this, the pager program sees an end of file and likewise also terminates.

Test Case #2: Endlessly Generating Words

```
hengnan@DESKTOP-HEBLQU8:~/ece357/p4/wordGen$ ./launcher
Accepted 416290 words, rejected 48830
AGU
OCK
RAE
TRON
NOEX
STE
PDS
IDC
SON
SLP
SIR
LCT
OXY
TIV
YEN
AGY
MCS
QKT
RTI
THUN
POC
PSV
CHP
---Press RETURN for more---
WCS
MYO
SGP
LADE
DTD
TED
BIK
PRN
RFZ
AFF
UMM
AJA
LAS
YEH
GIE
ABE
NAD
PPS
RLL
AGC
SMB
MANY
IVO
---Press RETURN for more---
Q
Child Process 2909 exited with status 0
wordsearch: Broken pipe - pager terminated
wordsearch: Number of words matched: 1999
Child Process 2908 exited with status 0
Child Process 2907 exited with status 0
hengnan@DESKTOP-HEBLQU8:~/ece357/p4/wordGen$
```

The figure above shows the output from the launcher program when the wordGen program endlessly generates words. The wordSearch program took a few minutes to process

the first 4096 words generated by wordGen. After every 23 lines, we see that the program prompts the user to hit return for more matches found by wordSearch. And once we hit Q, the pager program terminates. Because of this, the pipe between wordSearch and pager is broken, and the wordSearch program receives a SIGPIPE signal. This triggers our signal handler which causes wordSearch to print the number of words that were matched and then exit. Finally, because the read end of the pipe between wordGen and wordSearch has been closed, wordGen also receives a SIGPIPE signal causing wordGen to terminate as well.