Branch: master ▾ Find file Copy path

cs-35l / assignment5 / report.txt


prithvikannan added report

5053b6c on Nov 5

1 contributor

Raw Blame History

217 lines (185 sloc) 8.99 KB

```
1. wrote tr2b.c
I learned that in C the input count is passed in as an int argc and the inputs
are passed in as a char pointer argv. First I checked the input values to make
sure they were correct by checking if exactly 3 operands were given and that the
from and to string were the same length.

Then I verified that there were no duplicate letters, and built my dictionary
between the letters to translate.

Next I used getChar() to read a character, and if I had a translation for that
letter, I would apply it and use putChar(). Otherwise, I would just use
putChar() and go to the next character.

To compile, I used: gcc -std=c11 tr2b.c -o tr2b

2. wrote tr2u.c
Similar to tr2b.c, I started with input checking and then moved to creating my
dictionary.

However, instead of using getChar() and putChar(), I used a temporary buffer of
size 1 and the read() command to take in a single character. Then I would check
if that character could be translated and translate if necessary and then output
using the write() command.

To compile, I used: gcc -std=c11 tr2u.c -o tr2u

3. testing

man head
    Looked at the documentation for head, which takes the first x bytes of a
    file when used with the --bytes flag.
head --bytes=5000000 /dev/urandom > tester.txt
    ran this script to create a random file of 5000000 bytes.

man strace
    Looked at the documentation of strace, realized that I needed to pass
    -c flag for easy counting.

strace -c ./tr2b 'A' 'B' < tester.txt > result_b.txt
strace -c ./tr2u 'A' 'B' < tester.txt > result_u.txt
    Ran strace on buffered and unbuffered tr commands on my test input file.
    I set from to 'A' and to to 'B' arbitrarily, and piped the outputs to files.

    tr2b:
    % time     seconds  usecs/call     calls    errors syscall
    ------ ----------- ----------- --------- --------- ----------------
    0.00    0.000000           0         2           read
    0.00    0.000000           0         1           write
    0.00    0.000000           0         2           open
    0.00    0.000000           0         2           close
    0.00    0.000000           0         4           fstat
```

```
    0.00    0.000000           0        10           mmap
    0.00    0.000000           0         3           mprotect
    0.00    0.000000           0         1           munmap
    0.00    0.000000           0         1           brk
    0.00    0.000000           0         1         1 access
    0.00    0.000000           0         1           execve
    0.00    0.000000           0         1           arch_prctl
    ------ ----------- ----------- --------- --------- ----------------
    100.00    0.000000                    29         1 total


    tr2u:
    % time     seconds  usecs/call     calls    errors syscall
    ------ ----------- ----------- --------- --------- ----------------
    56.25    0.484831           0   5000000           write
    43.75    0.377132           0   5000002           read
    0.00    0.000000           0         2           open
    0.00    0.000000           0         2           close
    0.00    0.000000           0         2           fstat
    0.00    0.000000           0         8           mmap
    0.00    0.000000           0         3           mprotect
    0.00    0.000000           0         1           munmap
    0.00    0.000000           0         1           brk
    0.00    0.000000           0         1         1 access
    0.00    0.000000           0         1           execve
    0.00    0.000000           0         1           arch_prctl
    ------ ----------- ----------- --------- --------- ----------------
    100.00    0.861963              10000024         1 total


strace -c ./tr2b 'A' 'B' < tester.txt
strace -c ./tr2u 'A' 'B' < tester.txt
    Ran strace on buffered and unbuffered tr commands on my test input file.
    I set from to 'A' and to to 'B' arbitrarily, and had it output to the
    terminal.

    I got 29 system calls for tr2b and 10000024 for tr2u, just as above.



4. timing the runs
time ./tr2b 'A' 'B' < tester.txt > result_b.txt
time ./tr2u 'A' 'B' < tester.txt > result_u.txt
    Use the time command to keep track of how long the process took to run tr
    buffered and unbuffered.

    tr2b:
    real    0m0.004s
    user    0m0.000s
    sys     0m0.002s

    tr2u:
    real    0m9.232s
    user    0m1.377s
    sys     0m7.813s


ANALYSIS OF SFROB AND SFROBU

I created test files of various sizes using these commands
    head --bytes=0 /dev/urandom > zero.txt
    head --bytes=100 /dev/urandom > hundred.txt
    head --bytes=10000 /dev/urandom > tenthousand.txt

I tested performance of sfrobu.
For sfrobu, I would estimate (0.03-0.003)/(10000-100)x + 0.003
```

```
    time ./sfrobu < zero.txt
        real   0m0.003s
        user   0m0.001s
        sys    0m0.002s
    time ./sfrobu < hundred.txt
        real   0m0.003s
        user   0m0.002s
        sys    0m0.002s
    time ./sfrobu < tenthousand.txt
        real   0m0.030s
        user   0m0.012s
        sys    0m0.016s

I tested performance of sfrob.
For sfrob, I would estimate (0.004-0.003)/(10000-100)x + 0.003
    time ./sfrob < zero.txt
        real   0m0.003s
        user   0m0.001s
        sys    0m0.002s
    time ./sfrob < hundred.txt
        real   0m0.003s
        user   0m0.003s
        sys    0m0.001s
    time ./sfrob < tenthousand.txt
        real   0m0.004s
        user   0m0.000s
        sys    0m0.003s

The big O runtime of quicksort is O(nlogn), so our function is going to have
similar runtime.

I ran strace to look at the system calls for different sized inputs to my sfrobu

strace -c ./sfrobu < zero.txt
    % time     seconds  usecs/call     calls    errors syscall
    ------ ----------- ----------- --------- --------- ----------------
    20.00    0.000010           3         3           mprotect
    16.00    0.000008           2         4           fstat
    16.00    0.000008           8         1           munmap
    14.00    0.000007           2         4           brk
    12.00    0.000006           2         3           read
    12.00    0.000006           1         7           mmap
    6.00    0.000003           3         1           arch_prctl
    4.00    0.000002           1         2           close
    0.00    0.000000           0         2           open
    0.00    0.000000           0         1         1 access
    0.00    0.000000           0         1           execve
    ------ ----------- ----------- --------- --------- ----------------
    100.00    0.000050                    29         1 total
strace -c ./sfrobu < hundred.txt
    % time     seconds  usecs/call     calls    errors syscall
    ------ ----------- ----------- --------- --------- ----------------
    0.00    0.000000           0         3           read
    0.00    0.000000           0       101           write
    0.00    0.000000           0         2           open
    0.00    0.000000           0         2           close
    0.00    0.000000           0         4           fstat
    0.00    0.000000           0         7           mmap
    0.00    0.000000           0         3           mprotect
    0.00    0.000000           0         1           munmap
    0.00    0.000000           0         4           brk
    0.00    0.000000           0         1         1 access
    0.00    0.000000           0         1           execve
    0.00    0.000000           0         1           arch_prctl
    ------ ----------- ----------- --------- --------- ----------------
    100.00    0.000000                   130         1 total
```

```
strace -c ./sfrobu < tenthousand.txt
    % time     seconds  usecs/call     calls    errors syscall
    ------ ----------- ----------- --------- --------- ----------------
    99.96    0.037056           4     10000           write
    0.04    0.000013           2         8           brk
    0.00    0.000000           0         3           read
    0.00    0.000000           0         2           open
    0.00    0.000000           0         2           close
    0.00    0.000000           0         4           fstat
    0.00    0.000000           0         7           mmap
    0.00    0.000000           0         3           mprotect
    0.00    0.000000           0         1           munmap
    0.00    0.000000           0         1         1 access
    0.00    0.000000           0         1           execve
    0.00    0.000000           0         1           arch_prctl
    ------ ----------- ----------- --------- --------- ----------------
    100.00    0.037069                10033         1 total

The system calls for memory allocations are found by looking at the man page
for each of the syscalls from the stack trace.

man brk
    brk - change data segment size
    int brk(void *addr);
man mmap
    mmap - map or unmap files or devices into memory
    void *mmap(void *addr, size_t length, int prot, int flags,
                   int fd, off_t offset);
man munmap
    munmap - map or unmap files or devices into memory
           int munmap(void *addr, size_t length);

So brk is the equivalent of realloc, mmap is the equivalent of malloc, and
munmap is the equivalent of free.
```