CSE 344

System Programming

1st Assignment

Report

Author

Gökbey Gazi KESKİN

1901042631

Date

17.03.2022

Table of Contents

Algorithm

Step1 - main

First, main opens the file in read mode and checks for a lock. If the file is locked, it waits on the main until the file is unlocked by the other instances.

```
do{
    fcntl(fd, F_GETLK, &lock);
    if(lock.l_type!=F_UNLCK) write(1,"File is locked by another instance. Waitin
}while(lock.l_type!=F_UNLCK);
```

Then, main calls the readFile function which copies the file into a dynamically allocated buffer byte by byte. If the buffer gets full during the read operation readFile function reallocates the buffer. After the error checks, main calls the changeContent function, closes the file descriptor and frees the buffer.

Step 2 - changeContent

changeContent function parses the first command line argument into tokens (str1, str2, str1_alt). str1_alt token is used when multiple character matching ([ab]) is needed. It holds the alternative version of the string which will be replaced. Then 2 empty arrays (occurrences and sizes) are passed to contains function which fills them with indexes where str1 and str1_alt occurs in the buffer and their sizes. It handles all different situations (^, $, [], * etc.), parses accordingly and modifies the needed flags. Lastly it calls the replaceAndWrite function which modifies the buffer (removes the old occurrences and fills with new ones) and writes the new buffer to file. When changeContent function encounters the symbol ';' at the end of the operation, it sets the multipleFlag variable to 1 and repeats the whole process in a do-while loop.

Step 3- contains

Contains fuction traverses through the buffer (str1 parameter of the function) until the end.

```
while(str1[i]!='\0'){
```

When it encounters an occurrence of the first letter of str2 (or str2_alt), it keeps the current buffer index and runs through a couple of tests to determine if buffer contains str2 or not. During these tests, it also modifies str2 and str2_alt if the '*' operator is used.

```
isSubstr = str1[i]==str2[j] || (case_sensitive==0 && ((isUpper(str1[i]) && str1[i]==str2[j]-32)||(isLower(str1[i]) && str1[i]==str2[j]+32)));
isSubstr_alt = str1[i]==str2_alt[j] || (case_sensitive==0 && ((isUpper(str1[i]) && str1[i]==str2_alt[j]-32)||(isLower(str1[i]) && str1[i]==str2_alt[j]+32)));

index=-1;
if(isSubstr || isSubstr_alt){

    index = i;
```

If one of the tests fails, it sets the index to -1 again. Otherwise (if index is not -1) it adds the index to occurrence array and the size of the occurrence to sizes array (each occurrence may have different sizes since * operator modifies the size).

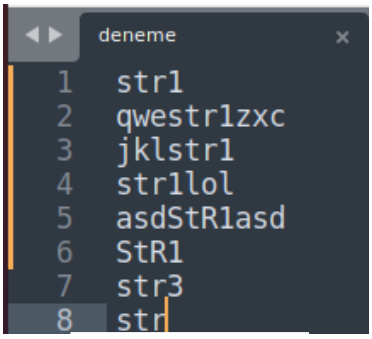
```
if(index != -1){
    i--;
    isValid = (lineStartFlag==0
    if(isValid){
        ocurrences[k++]=index;

    }
}
i++;
```

Step 4 - replaceAndWrite

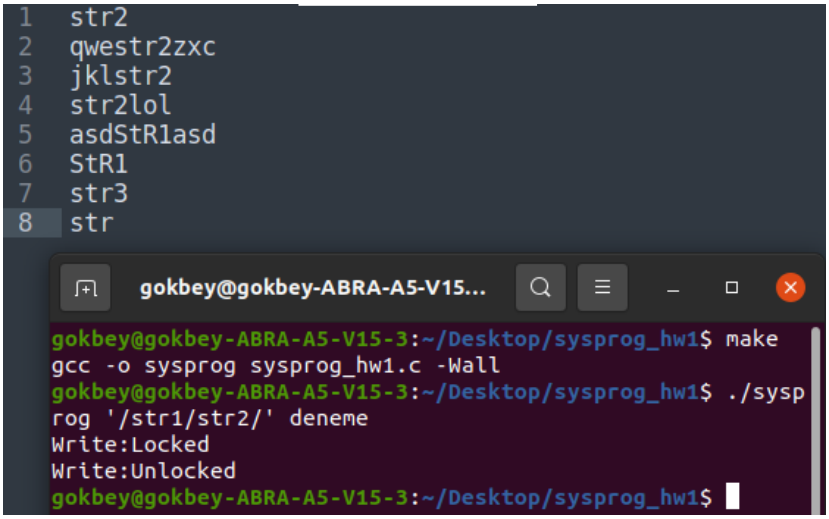
replaceAndWrite function opens the file again in write mode and sets a lock. Then, it traverses the occurrences array and replaces the old words in the buffer with new words using their occurrence index and size. It also shifts the buffer right if the new text is longer then the old text and shifts it to left if it is vice versa. Finally, it writes the buffer to file, unlocks the file, and closes file descriptor.

Test Cases

Case 1 – Basic String Replacement (Case Sensitive & Insensitive)



Input



Case Sensitive Output & Terminal

Case Insensitive Output & Terminal

Case 2 – Multiple Operations

deneme

```
1  str1
2  sTr1
3  asdstr1qwe
4  ewqStR1dsa
5  str3
6  asdstr3asd
7  sTr3
```

Input 1

deneme | sysprog_hw1.c

```
1  str2
2  str2
3  asdstr2qwe
4  ewqstr2dsa
5  str4
6  asdstr4asd
7  sTr3
```

gokbey@gokbey-ABRA-A5-V15-3: ~/Desktop/sysprog_hw1

```
gokbey@gokbey-ABRA-A5-V15-3:~/Desktop/sysprog_hw1$ ./sysprog '/str1/str2/i;/str3/str4/' deneme
Write:Locked
Write:Unlocked
Write:Locked
Write:Unlocked
gokbey@gokbey-ABRA-A5-V15-3:~/Desktop/sysprog_hw1$
```

Output & Terminal 1

```
1  str1
2  asdstr1
3  qwestr1dsa
4  sTr1
5  kjuStR1
6  StR1pkl
7  yhnsTr1jmn
8
9  str2
10 asdstr2
11 qwestr2dsa
12 sTr2
13 kjuStR2
14 StR2pkl
15 yhnsTr2jmn
16
17 str3
18 asdstr3
19 qwestr3dsa
20 sTr3
21 kjuStR3
22 StR3pkl
23 yhnsTr3jmn
```

Input 2

```
1  Gokbey
2  asdGokbey
3  qweGokbeydsa
4  Gokbey
5  kjuGokbey
6  Gokbeypkl
7  yhnGokbeyjmn
8
9  Gazi
10 asdGazi
11 qweGazidsa
12 sTr2
13 kjuStR2
14 StR2pkl
15 yhnsTr2jmn
16
17 KESKIN
18 asdKESKIN
19 qweKESKINdsa
20 KESKIN
21 kjuKESKIN
22 KESKINpkl
23 yhnKESKINjmn
```

gokbey@gokbey-ABRA-A5-V15...

```
Write:Unlocked
186
Write:Locked
200
Write:Unlocked
gokbey@gokbey-ABRA-A5-V15-3:~/Desktop/sysprog_hw1$ make
gcc -o sysprog sysprog_hw1.c -Wall
gokbey@gokbey-ABRA-A5-V15-3:~/Desktop/sysprog_hw1$ ./sysp
rog '/str1/Gokbey/i;/str2/Gazi/;/str3/KESKIN/i' deneme
Write:Locked
Write:Unlocked
Write:Locked
Write:Unlocked
Write:Locked
Write:Unlocked
gokbey@gokbey-ABRA-A5-V15-3:~/Desktop/sysprog_hw1$
```

Output & Terminal 2

Case 3 – Multiple Character Matching

```
1  str1
2  ztr1
3  asdstr1weq
4  gtdtjztr1grh
```

Input 1

```
1  str2
2  str2
3  asdstr2weq
4  gtdtjstr2grh
5
```

```
gokbey@gokbey-ABRA-A5-V15-3: ~/Desktop/sysprog_hw1
gokbey@gokbey-ABRA-A5-V15-3:~/Desktop/sysprog_hw1$ ./sysprog '/[zs]tr1/str2/' deneme
Write:Locked
Write:Unlocked
```

Output & Terminal 1

```
1  muhammet
2  muhammed
3  asdasfmuhammet
4  fsgemgmuhammedgslgms
```

Input 2

```
1  gokbey
2  gokbey
3  asdasfgokbey
4  fsgemggokbeygslgms
```

```
gokbey@gokbey-ABRA-A5-V15-3: ~/Desktop/sysprog_hw1
gokbey@gokbey-ABRA-A5-V15-3:~/Desktop/sysprog_hw1$ ./sysprog '/muhamme[td]/gokbey/'
deneme
Write:Locked
Write:Unlocked
```

Output & Terminal 2

Case 4 – Matching Line Starts & Ends

```
1  john doe
2  placeholder john doe
3  john doe placeholder
4  placeholder john doe placeholder
5
```

Input

```
1  gokbey
2  placeholder john doe
3  gokbey placeholder
4  placeholder john doe placeholder
```

```
gokbey@gokbey-ABRA-A5-V15-3: ~/Desktop/sysprog_hw1
gokbey@gokbey-ABRA-A5-V15-3:~/Desktop/sysprog_hw1$ ./sysprog '/^john doe/gokbey/
' deneme
Write:Locked
Write:Unlocked
```

Output & Terminal for line start

```
1  gokbey
2  placeholder gokbey
3  john doe placeholder
4  placeholder john doe placeholder
```

```
gokbey@gokbey-ABRA-A5-V15-3: ~/Desktop/sysprog_hw1
gokbey@gokbey-ABRA-A5-V15-3:~/Desktop/sysprog_hw1$ ./sysprog '/john doe$/gokbey/
' deneme
Write:Locked
Write:Unlocked
```

Output & Terminal for line end

Case 5 – Multiple Occurrence Matching

```
1   0 ocurrence:
2   sr1
3   1 ocurrence:
4   str1
5   more ocurrence:
6   stttttttttr1
7   stttttttttttttttttr1
8
9   asdstr1ads
10  sadstttr1
11  sttttr1grr
```

Input

```
1   0 ocurrence:
2   CSE344
3   1 ocurrence:
4   CSE344
5   more ocurrence:
6   CSE344
7   CSE344
8
9   asdCSE344ads
10  sadCSE344
11  CSE344grr
```

```
gokbey@gokbey-ABRA-A5-V15-3: ~/Desktop/sysprog_hw1
gokbey@gokbey-ABRA-A5-V15-3:~/Desktop/sysprog_hw1$ ./sysprog '/st*r1/CSE344/' de
neme
Write:Locked
Write:Unlocked
```

Output & Terminal

Arbitrary Combinations of Test Cases

```
1   sttttr1
2   sddddr1
3   str1
4   sdr1
5   sr1
6   placeholder sddddr1 placeholder
7   placeholder sttttr1 placeholder
8   sddddddddddr1
9   stttttttttttttttttttttttr1
```

Input 1

```
1   CSE344
2   CSE344
3   CSE344
4   CSE344
5   CSE344
6   placeholder CSE344 placeholder
7   placeholder CSE344 placeholder
8   CSE344
9   CSE344
```

```
gokbey@gokbey-ABRA-A5-V15-3: ~/Desktop/sysprog_hw1
gokbey@gokbey-ABRA-A5-V15-3:~/Desktop/sysprog_hw1$ ./sysprog '/s[td]*r1/CSE344/
' deneme
Write:Locked
Write:Unlocked
```

Output & Terminal 1

```
1   john doe
2   johhhn doe
3   JoHn DoE
4   JoAn Doe
5   Joan Doe
6   placeholder john doe
7   placeholder joan doe
8   placeholder Joan Doe
9   john doe placeholder
10  Joan Doe placeholder
11  placeholder john doe placeholder
12  placeholder Joan Doe placeholder
```

Input 2

```
1   gokbey
2   gokbey
3   JoHn DoE
4   gokbey
5   gokbey
6   xxxxxx john doe
7   xxxxxx joan doe
8   xxxxxx Joan Doe
9   gokbey xxxxxx
10  gokbey xxxxxx
11  xxxxxx john doe xxxxxx
12  xxxxxx Joan Doe xxxxxx
```

```
gokbey@gokbey-ABRA-A5-V15-3: ~/Desktop/sysprog_hw1
gokbey@gokbey-ABRA-A5-V15-3:~/Desktop/sysprog_hw1$ ./sysprog '/^jo[ah]*n doe/gok
bey/i;/placeholder/xxxxxx/' deneme
Write:Locked
Write:Unlocked
Write:Locked
Write:Unlocked
```

Output & Terminal 2

```
1  Windows closed
2  windoWs closed
3  WiNdowz closeD
4  closed windowz
5  closeD WiNdOwz
6
7  closedwrt
8  asdcloseD
```

Input 3

```
1  Linux open
2  Linux open
3  Linux open
4  closed windowz
5  closeD WiNdOwz
6
7  closedwrt
8  asdopen
```

```
gokbey@gokbey-ABRA-A5-V15-3: ~/Desktop/sysprog_hw1
gokbey@gokbey-ABRA-A5-V15-3:~/Desktop/sysprog_hw1$ ./sysprog '/^Window[sz]*/Linu
x/i;/close[dD]$/open/' deneme
Write:Locked
Write:Unlocked
Write:Locked
Write:Unlocked
```

Output & Terminal 3

File Lock Test

```
1  str1
2
3  str3
4
```

Input

Input of terminal 1: ‘/str1/str2/’ Input of terminal 2: ‘/str3/str4/’

I used sleep function to test file lock feature. Process 1 sleep for 5 seconds after file lock and this gives me time to start process 2 on the other terminal. (I commented out the sleep function after test)