TUBA TOPRAK

161044116 -Hw1

SYSTEM PROGRAMMING

Content

Homework Overview

Part1 is written inside the appendMeMore.c file.

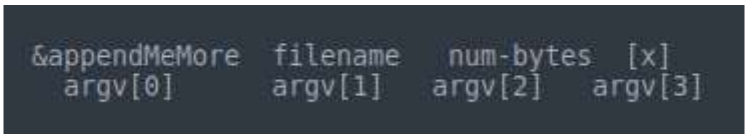
Part2 and Part3 are written into the part2_3.c file.

An extra txt has not been sent since all parts create their own files.

MAKEFILE compiles but does not run two .c files.

A file named dup_and_dup2.txt was created in part2. This file was used in part3.

Part1 Overview



```
&appendMeMore  filename   num-bytes  [x]
  argv[0]      argv[1]   argv[2]   argv[3]
```

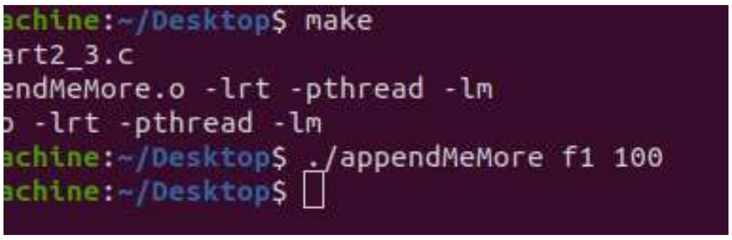
After determining the locations of the arguments, it was checked whether missing or excess arguments were entered. The rest of the assignment is built with if else blocks. If the x argument is not entered, the file is opened with the O_APPEND flag and letters are added to the file up to the entered byte.the file is overwritten if it exists, otherwise a new file is created. The O_append flag adds to the end of the file. In order to see this, the letters “t” and “u” were suppressed to the file and "tutututut..." was printed. The open() write() functions were tested to work properly. If there is an error, the error is written with perror. opened file is closed.

If the X argument is entered, the O_append flag is removed, the file is overwritten if it exists, otherwise a new file is created. The same steps were continued and the file was written with the lseek() and write() function. The letters “b” and “a” were used to check that it was written towards the end of the file. And when the file was run, the text "bababa...." was seen.The open() write() functions were tested to work properly. If there is an error, the error is written with perror. opened file is closed.

Tests

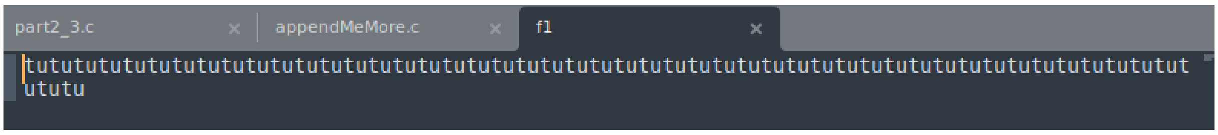
$ appendMeMore f1 num-bytes

First, a non-existent filename was tried.



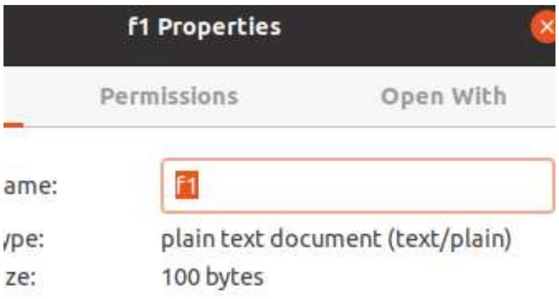
```
achine:~/Desktop$ make
art2_3.c
endMeMore.o -lrt -pthread -lm
o -lrt -pthread -lm
achine:~/Desktop$ ./appendMeMore f1 100
achine:~/Desktop$ 
```

File is expected to start with letter “t”.



```
tutututututututututututututututututututututututututututututututututututututututututututututut
ututu
```

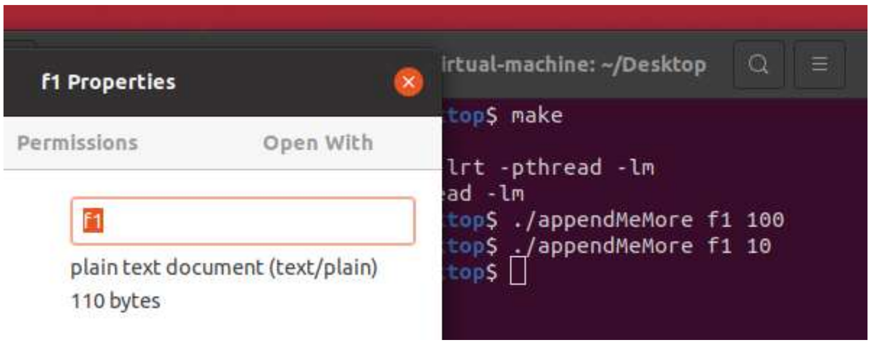
The number of bytes entered has been checked.



```
ame:  f1
ype:  plain text document (text/plain)
ze:   100 bytes
```

When processing an existing file:

The number of bytes entered has been checked.



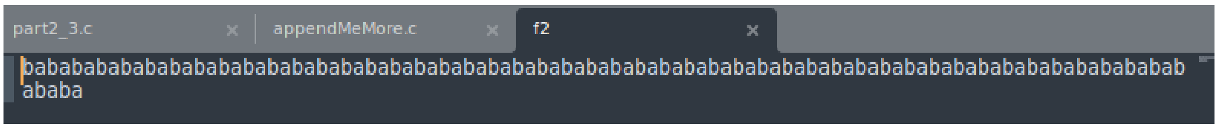
```
f1
plain text document (text/plain)
110 bytes
```

```
top$ make

lrt -pthread -lm
ad -lm
top$ ./appendMeMore f1 100
top$ ./appendMeMore f1 10
top$ 
```

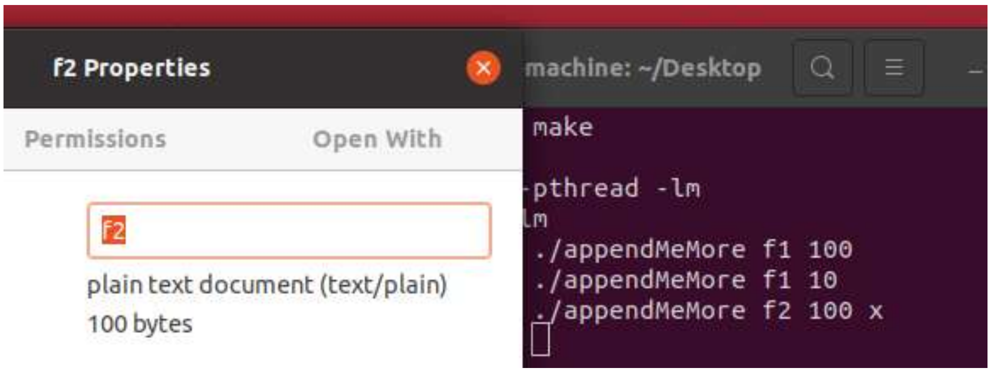
$ appendMeMore f2 numbytes x

First, a non-existent filename was tried. File is expected to start with letter "b".



```
bababababababababababababababababababababababababababababababababababababababababababababababab
ababa
```

The number of bytes entered has been checked.

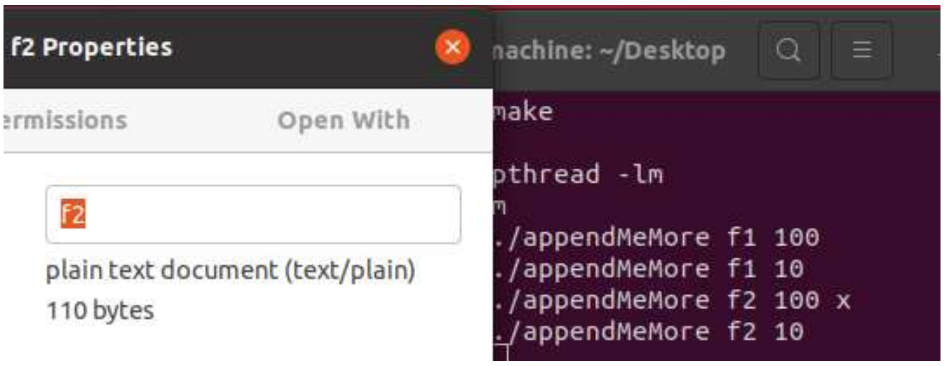


```
f2
plain text document (text/plain)
100 bytes
```

```
make

-pthread -lm
lm
./appendMeMore f1 100
./appendMeMore f1 10
./appendMeMore f2 100 x
```

When processing an existing file and The number of bytes entered has been checked.



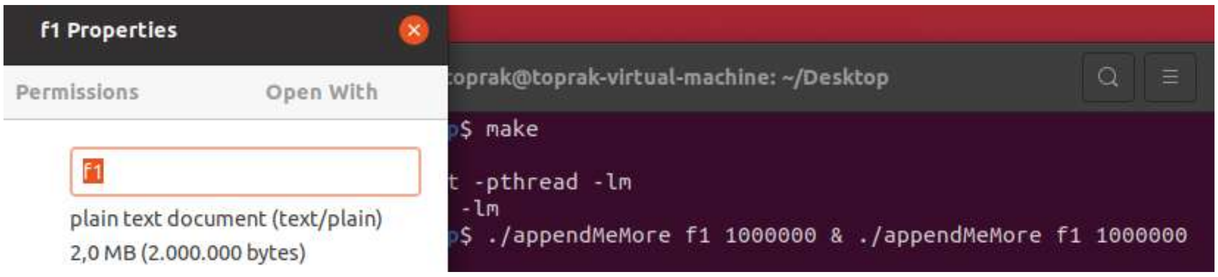
```
f2
plain text document (text/plain)
110 bytes
```

```
make

pthread -lm
m
./appendMeMore f1 100
./appendMeMore f1 10
./appendMeMore f2 100 x
./appendMeMore f2 10
```

$ appendMeMore f1 1000000 & appendMeMore f1 1000000

Deleted f1 and f2 file created earlier. It was recompiled from the program and the arguments were entered as desired. The file was created and byte checked.



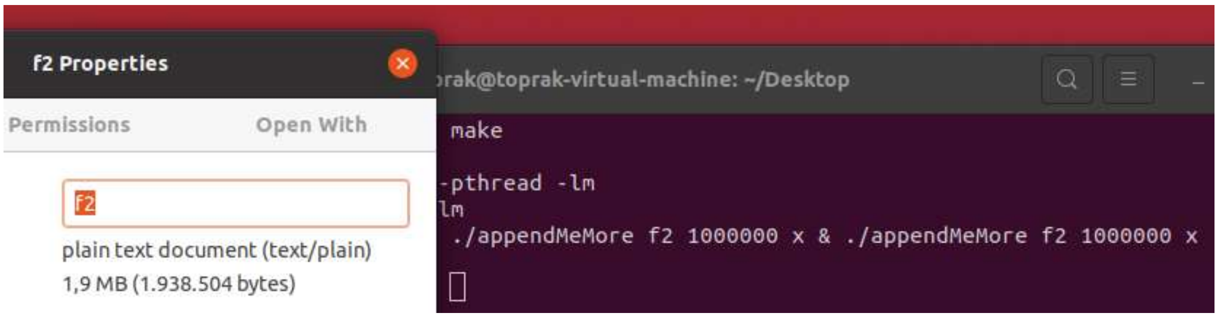
```
f1
plain text document (text/plain)
2,0 MB (2.000.000 bytes)
```

```
oprak@toprak-virtual-machine: ~/Desktop
p$ make

t -pthread -lm
 -lm
p$ ./appendMeMore f1 1000000 & ./appendMeMore f1 1000000
```

$ appendMeMore f2 1000000 x & appendMeMore f2 1000000 x

It was recompiled from the program and the arguments were entered as desired. The file was created and byte checked.

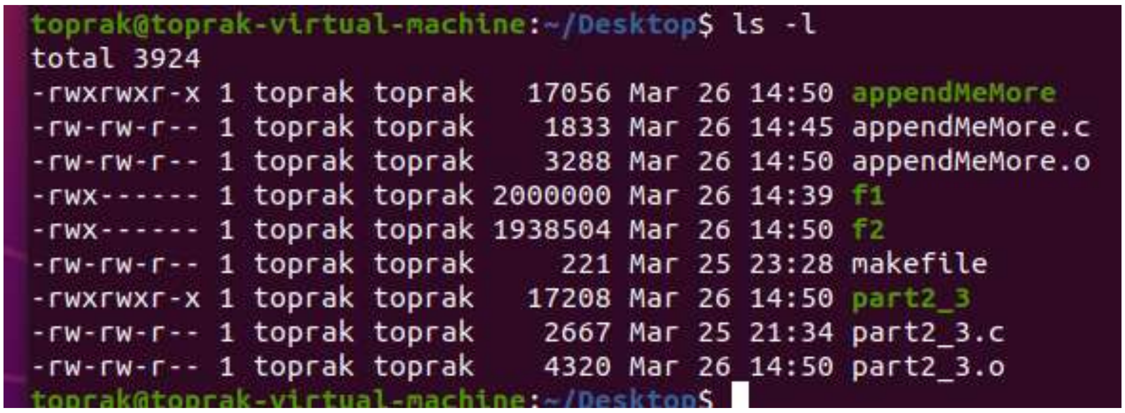


```
f2
plain text document (text/plain)
1,9 MB (1.938.504 bytes)
```

```
rak@toprak-virtual-machine: ~/Desktop
 make

-pthread -lm
lm
 ./appendMeMore f2 1000000 x & ./appendMeMore f2 1000000 x
```

LS -L



```
toprak@toprak-virtual-machine:~/Desktop$ ls -l
total 3924
-rwxrwxr-x 1 toprak toprak   17056 Mar 26 14:50 appendMeMore
-rw-rw-r-- 1 toprak toprak    1833 Mar 26 14:45 appendMeMore.c
-rw-rw-r-- 1 toprak toprak    3288 Mar 26 14:50 appendMeMore.o
-rwx------ 1 toprak toprak 2000000 Mar 26 14:39 f1
-rwx------ 1 toprak toprak 1938504 Mar 26 14:50 f2
-rw-rw-r-- 1 toprak toprak     221 Mar 25 23:28 makefile
-rwxrwxr-x 1 toprak toprak   17208 Mar 26 14:50 part2_3
-rw-rw-r-- 1 toprak toprak    2667 Mar 25 21:34 part2_3.c
-rw-rw-r-- 1 toprak toprak    4320 Mar 26 14:50 part2_3.o
toprak@toprak-virtual-machine:~/Desktop$
```

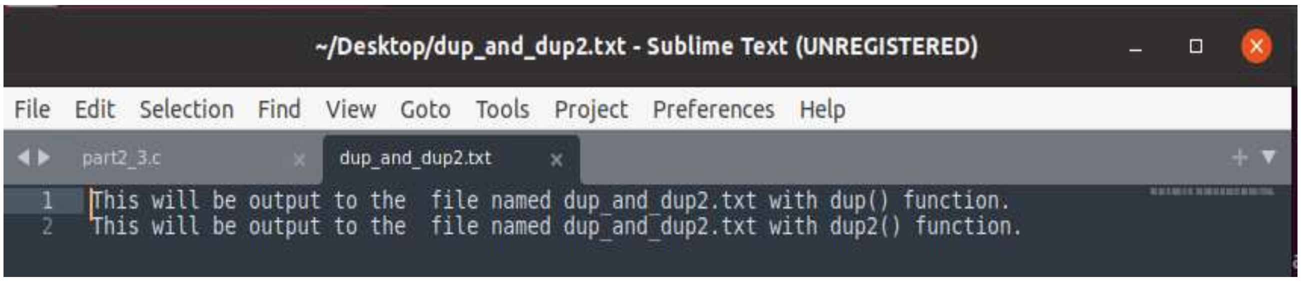
I found that there is a difference between the files when they are run. This is because with the lseek() function, the root of the file shows the end. Data loss occurs because the program does not run at the same time as writing while searching for the end of the file.

Part 2 and Part 3 Overview

A file was created in part2. When the dup and dup2 functions are implemented, this file is written with dup() and dup2(). Later in Part3, this file was read with the dup() function. The read() and write() functions have been checked for errors. Errors are indicated with the perror() function for possible errors.

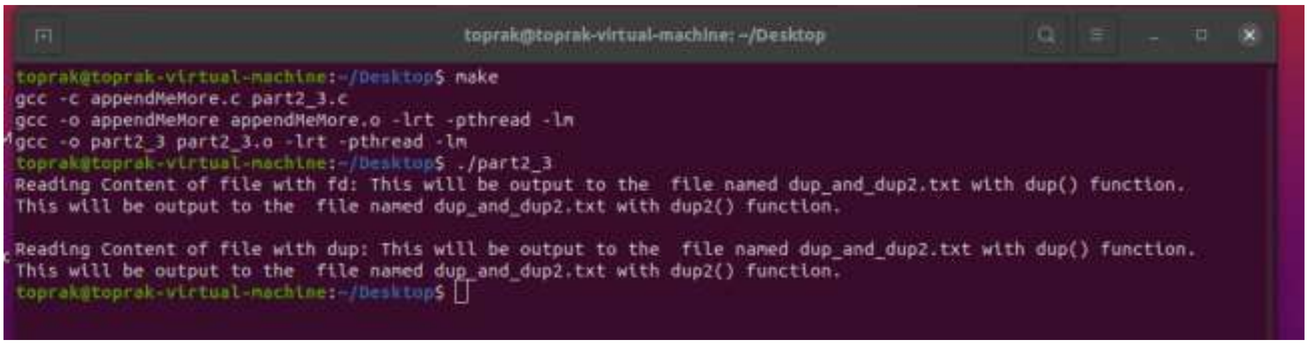
Tests

When Part2 is applied, you see the contents of the file.


~/Desktop/dup_and_dup2.txt - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
part2_3.c
dup_and_dup2.txt

```
This will be output to the  file named dup_and_dup2.txt with dup() function.
This will be output to the  file named dup_and_dup2.txt with dup2() function.
```

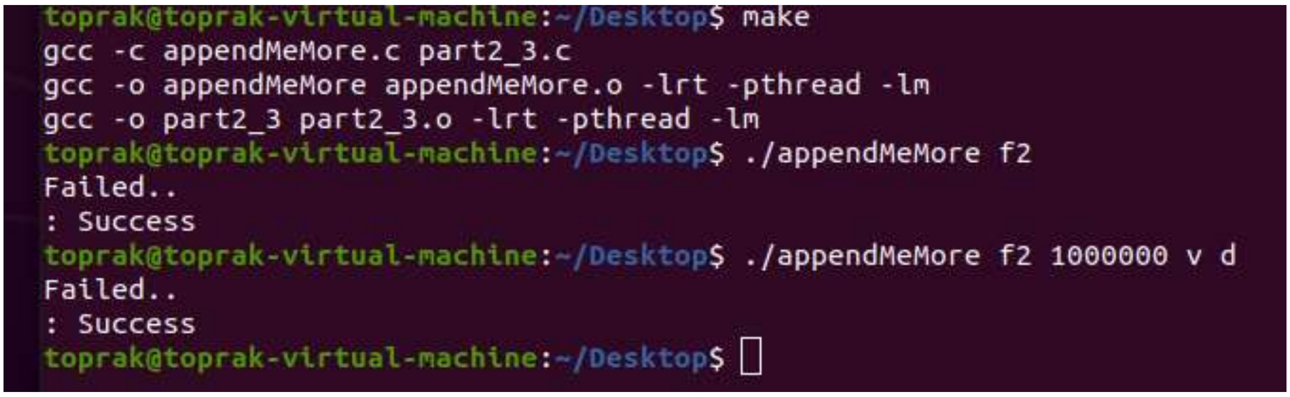
When part3 applied, the contents read to the console are displayed.



```
toprak@toprak-virtual-machine:~/Desktop$ make
gcc -c appendMeMore.c part2_3.c
gcc -o appendMeMore appendMeMore.o -lrt -pthread -lm
gcc -o part2_3 part2_3.o -lrt -pthread -lm
toprak@toprak-virtual-machine:~/Desktop$ ./part2_3
Reading Content of file with fd: This will be output to the  file named dup_and_dup2.txt with dup() function.
This will be output to the  file named dup_and_dup2.txt with dup2() function.

Reading Content of file with dup: This will be output to the  file named dup_and_dup2.txt with dup() function.
This will be output to the  file named dup_and_dup2.txt with dup2() function.
toprak@toprak-virtual-machine:~/Desktop$
```

For part1



```
toprak@toprak-virtual-machine:~/Desktop$ make
gcc -c appendMeMore.c part2_3.c
gcc -o appendMeMore appendMeMore.o -lrt -pthread -lm
gcc -o part2_3 part2_3.o -lrt -pthread -lm
toprak@toprak-virtual-machine:~/Desktop$ ./appendMeMore f2
Failed..
: Success
toprak@toprak-virtual-machine:~/Desktop$ ./appendMeMore f2 1000000 v d
Failed..
: Success
toprak@toprak-virtual-machine:~/Desktop$
```