1. Introduction
   1. Project overview

The idea of the project was to create a program that reverses the given text. For example, if the program was given a file with the following text:

Hello

this

is

a file

The output would then be

a file

is

this

Hello

* 1. Features

The program could be invoked with one of the following ways:

prompt> ./reverse

prompt> ./reverse input.txt

prompt> ./reverse input.txt output.txt

The first way the program reads the input from standard input and prints the output to standard output. The second way the program reads the input from a file and prints the output to standard output. The third way the program reads the input from a file and prints the output to a file.

The program has following assumptions:

* String length: There is no assumption of how long a line is.
* File length: There is no assumption of how long a file is.

The program handless the following errors:

* Input is the same as output: The program gives an error message “Input and output file must differ” and exit with return code 1 if the input and output files have same name or if they are the same file (hardlinked).
* Invalid files: If the input or output files fail to open, the program gives an error “error: cannot open file 'file.txt'” and exit with return code 1.
* Malloc fails: If the memory allocation fails, the program prints an error message “malloc failed” and exits with a return code 1.
* Too many arguments passed to program: if the program is run with too many arguments the program prints “usage: reverse <input> <output>” and exit with return code 1.

The error messages are printed to the screen using *fprintf()* and the error message is sent to *stderr*.

1. Implementation

The program is based on a linked list. The input is read line by line to a linked list:

A computer screen shot of text

Description automatically generated

where *addNode()* adds the line to a linked list:

A screen shot of a computer program

Description automatically generated

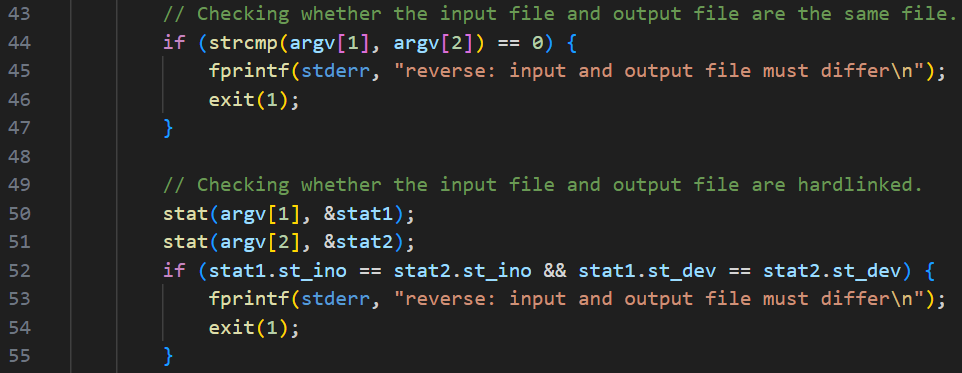
To print the lines in reversed order the program goes through the linked list to the last node and prints the lines from the last node to the first node:

A computer screen shot of text

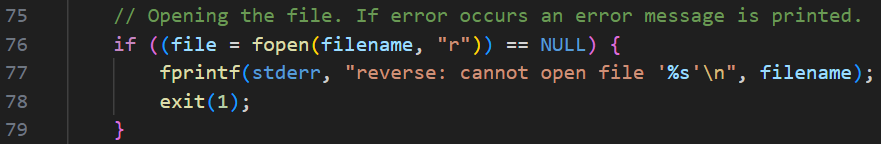
Description automatically generated

How the errors are handled:

* Input is the same as output: The program checks if the input and output files have the same name or if they have the same inode number and same device ID (=hardlinked).



* Invalid files:

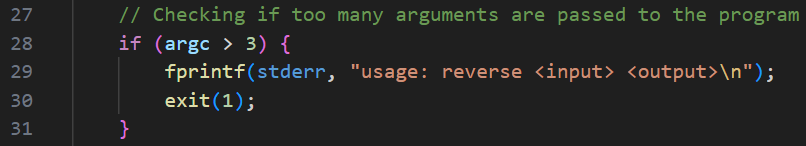


* Malloc fails:

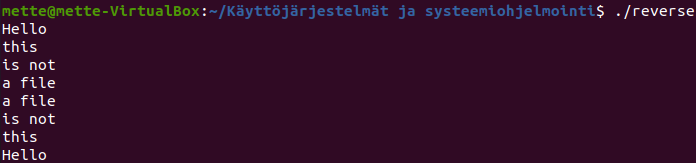
A black screen with colorful text

Description automatically generated

* Too many arguments passed to program: The program checks if there are more than three arguments given to the program. If so, the error message is printed.



The output of the program when the program is invoked with ./reverse:



Example of an error message when too many arguments are given to the program:

