Project 1 Warmup to C and Unix

The program reverse.c reverses the input it gets line by line. There are 3 ways of invoking the program first with no arguments where we read input from stdin and output the reversed result to stdout. Then with one file which is an input file where we read the contents of the file, reverse it and paste the results to stdin. Then finally with 2 arguments which are input and output files where we again read the input file but now instead of writing to stout, we write to the output file.

Working principle of the program

How my reverse program is implemented is quite simple. The program consists mainly of a singly linked list that is NULL terminated. We read from the input stream in question (File or stdin) and then we write the contents line by line to the linked list in the following way. Each new line that is added to the list is added to the head of the list in this way as a result we get a list that from the head to the NULL terminator is in reverse order compared to the input. So, to speak the last line of the input will be the first element of the list already after construction. After the construction of the list, we write the contents of the list to the desired output.

What I did written shortly is that first I programmed the linked list and then which I needed some help in for remembering how to doo a linked list and how to handle them these resources were helpful for that ("C - Pointer to Pointer (Double Pointer)," 22:05:52+00:00) ("Linked List Data Structure," n.d.) (Codes, 2020). The exact points where they impacted my source codes can be seen from the source code. Then I did the data parsing from the input stream and the handling the outputs and lastly error handling.

Functions

The program consists of the following functions:

Node *createNode(char *data)

Takes in a pointer to character data and creates a Node containing the data and returns the new Node.

void addNode(Node **head, Node *node)

Takes in a double pointer to the head of the linked list and a node to add to the list. Adds the new node to the head of the list.

void freeMemory(Node *head)

Frees the memory associated with the head of the list.

void parseFile(char *fileName, Node **head)

Parses the input file line by line and adds the contents to the linked list associated with the head parameter.

void parseStdIn(Node **head)

Does the same as parseFile except the iinput is now stdin and not a file.

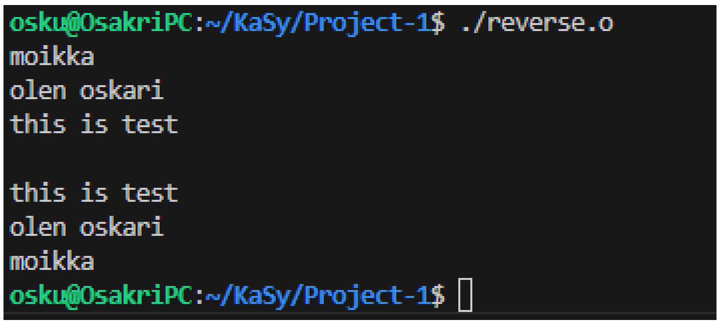
void writeFile(char *fileName, Node *head)

Writes the contents oof the list that starts at the parameter head and writes it to the file named the parameter filename.

void printList(Node *head)

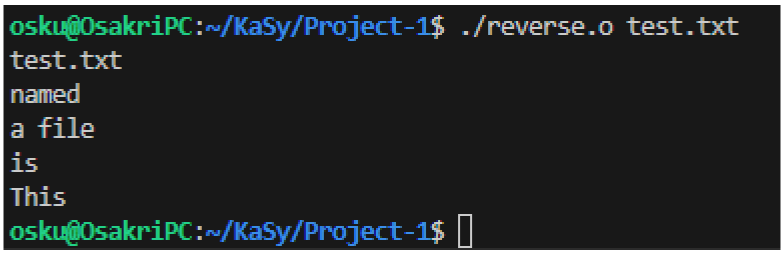
Prints the contents of the list starting at parameter head to stdin.

Screenshots of functional versions of the program.



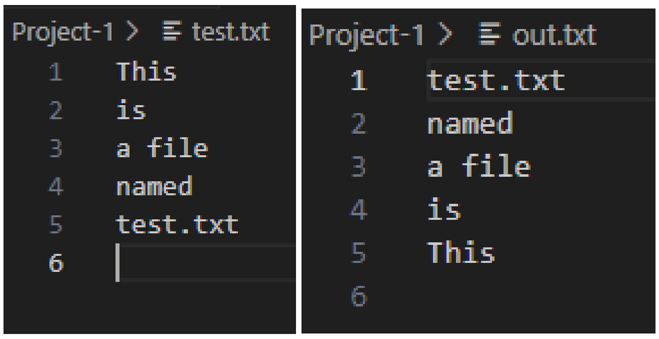
```
osku@OsakriPC:~/KaSy/Project-1$ ./reverse.o
moikka
olen oskari
this is test

this is test
olen oskari
moikka
osku@OsakriPC:~/KaSy/Project-1$ 
```

Picture 1. Running the program with no arguments



```
osku@OsakriPC:~/KaSy/Project-1$ ./reverse.o test.txt
test.txt
named
a file
is
This
osku@OsakriPC:~/KaSy/Project-1$ 
```

Picture 2. Running the program with one argument test.txt, contents of test.txt provided in picture 3



```
Project-1 > test.txt
This
is
a file
named
test.txt

```

```
Project-1 > out.txt
test.txt
named
a file
is
This

```

Picture 3. Contents of test.txt and out.txt after running the program with 2 arguments test.txt and out.txt

In picture 1. The program is run with no arguments, and it reads input that I wrote to stdin and produces the reversed output. Picture 2 shows execution of the program with the file test.txt shown in picture 3 on the left. The program reads the file and produces output to stdout. Then finally I ran the program with arguments test.txt and out.txt like this "./reverse.o test.txt out.txt" and the output file out.txt contents is shown in picture 3 on the right.

References

C - Pointer to Pointer (Double Pointer) [WWW Document], 22:05:52+00:00. . GeeksforGeeks. URL https://www.geeksforgeeks.org/c-pointer-to-pointer-double-pointer/ (accessed 5.10.25).

Codes, C., 2020. Building a Linked List of Strings. Stack Overflow.

Linked List Data Structure [WWW Document], n.d. URL https://www.programiz.com/dsa/linked-list (accessed 5.10.25).