

Developer Documentation Morse Code

Introduction

This program serves as a Morse Code encoding and decoding tool, capable of accepting input from either the standard input or files. It can encode English text or decode Morse code, maintaining a translation history and providing statistics on the input and output texts. The program utilizes a binary tree for efficient Morse code decoding and incorporates file handling capabilities, allowing users to process data from external files for both encoding and decoding operations.

Solution

The program consists of several modules: ``main.c``, ``encode.h``, ``decode.h``, and ``statistics.h``.

morse.c

This module contains the main entry point for the program. It handles command-line arguments, processes user input, calls the appropriate functions, and maintains a translation history.

encode.h

This module provides functions for encoding English text to Morse code. It includes the "encode" function, which takes English text as input and produces the corresponding Morse code. The module uses a predefined mapping of English letters to Morse code representations.

decode.h

This module handles Morse code decoding using a binary tree data structure. It defines functions to create, traverse, and drop the Morse code tree. The "decode" function takes the Morse code input and produces the corresponding English text.

statistics.h

This module provides the possibility to analyze the input and output and calculate the statistics of characters. In the case of English text, it calculates statistics for English letters. In the case of Morse code, it analyzes the occurrence of dots and dashes.

File Handling:

The program utilizes file handling for encoding or decoding purposes. During these operations, the manipulated original text is stored in a file named "output.txt". This file is then used to display the text to the user. Additionally, the history of translations is recorded in two separate files: one for English text to Morse code translations (e2m) and another for Morse code to English text translations (m2e).

Dynamic Memory Allocation:

This program actively employs dynamic memory allocation. Two functions are utilized for obtaining input—one for reading long lines from standard input (stdin) and another for reading from a file. These functions dynamically allocate memory for the input, ensuring that only the necessary amount of memory is used. Unlike fixed-size allocations, these functions incrementally increase the size of the string with each character entered or read. Input is accepted until an EOF signal is encountered.

Data Structures

Binary Tree

The binary tree efficiently decodes Morse code, with each node representing the Morse code representation of English letters. The "decode" function traverses this tree based on the input Morse code to decode the corresponding English text.

Algorithms

Encoding Algorithm

The encoding algorithm in `encode.h` involves iterating over each character in the input English text, finding its Morse code equivalent based on the predefined mapping. It supports lower case and upper case letters. The result is printed to the standard output and stored in the translation history.

Decoding Algorithm

The decoding algorithm in `decode.h` uses a binary tree data structure to efficiently decode Morse code. The `decode` function traverses the Morse tree based on the input Morse code, and the decoded characters are printed to standard output and stored in the translation history.

Statistics Algorithm

The statistics algorithm in `statistics.h` counts occurrences of dots, dashes, and characters in English text.

List of Functions

encode.h

void encode(char *text):

encodes English text to Morse code.

decode.h

Recursive functions:

- **static void decode_rec(node *tree, const char *text, FILE *output):**

it goes through the text, and points where to and what to do if it meets '.', '_', *space*, '\0'

- **static void insert_rec(node **tree, char letter, const char *text):**

recursive function, which analyzes the text, and goes (in the tree) to left(if .), right(if _) or prints the letter needed (if the space which delimites the letters is encountered)

- **void drop_tree_rec(node *root):**

deallocates memory by traversing a binary tree.

Main-decode functions

- **void decode(const char *text, node *root):**

decodes Morse code to English text using a binary tree.

- **void create_tree(node **root):**

creates the binary tree for Morse code decoding.

- **void drop_tree(node *root):**

drops the binary tree to free memory.

statistics.h

- **void statistics(char *text):**

analyzes the statistics of the input text.

morse.c

- **int main(int argc, char *argv[]):**

main entry point of the program. Handles command-line arguments, processes input, and calls encoding, decoding, or history functions.

- **char *read_long_line_from_stdin(char mode):**

reads a long line from standard input.

- **char *read_long_line_from_file(char mode, char *filename):**

reads a long line from a file.

- **void view_or_delete_history(const char *filename):**

views or deletes the translation history, providing the user the option to delete or keep the history.

- **void write_to_history(char *filename, char *str):**

writes the translation history to a file.

- **FILE *open_file(const char *filename, const char *mode):**

opens a file with the specified mode, checking for existence or whether the file was created.

Function Interfaces

From the command line, the program is called as follows:

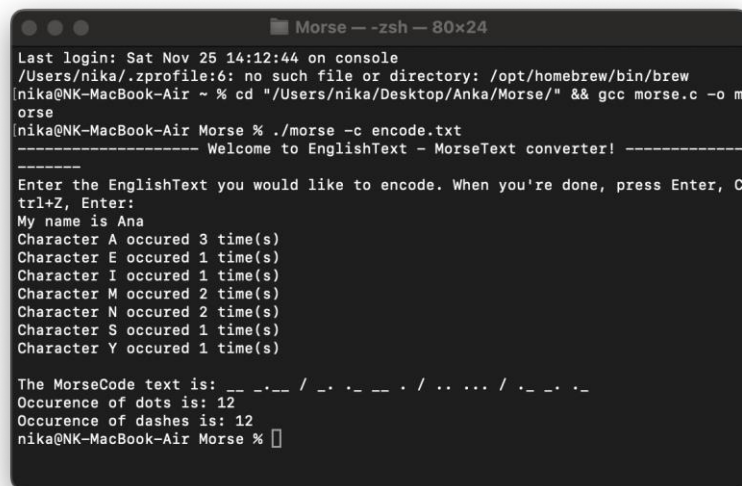
```
morse.c <switch> <filename>
```

Switches:

- `-c` for encoding
- `-d` for decoding
- `-h` for viewing history

If the user wants to encode personal input, use `encode.txt` as the filename:

```
morse.c -c encode.txt
```



```
Morse - zsh - 80x24
Last login: Sat Nov 25 14:12:44 on console
/Users/nika/.zprofile:6: no such file or directory: /opt/homebrew/bin/brew
[nika@NK-MacBook-Air ~ % cd "/Users/nika/Desktop/Anka/Morse/" && gcc morse.c -o m]
morse
[nika@NK-MacBook-Air Morse % ./morse -c encode.txt]
----- Welcome to EnglishText - MorseText converter! -----
-----
Enter the EnglishText you would like to encode. When you're done, press Enter, C
trl+Z, Enter:
My name is Ana
Character A occurred 3 time(s)
Character E occurred 1 time(s)
Character I occurred 1 time(s)
Character M occurred 2 time(s)
Character N occurred 2 time(s)
Character S occurred 1 time(s)
Character Y occurred 1 time(s)

The MorseCode text is: __ _ . _ _ / _ . _ _ . / . . . . / _ . _ . _
Occurrence of dots is: 12
Occurrence of dashes is: 12
nika@NK-MacBook-Air Morse %
```

Here is entered a personalized input. The output shows the Morse code equivalent of the input and the statistics of both input and output.

If the user wants to encode an existing text, use the proper filename:

```
morse.c -c <filename>
```

[illegible]

Here is encoded an already existing file. The statistics is presented for both input and output.

If the user wants to decode personal input, use `decode.txt` as the filename:

```
morse.c -d decode.txt
```

```
Morse -- -zsh -- 94x27
nika@NK-MacBook-Air Morse % ./morse -d decode.txt
----- Welcome to MorseText - EnglishText converter! -----
Enter the MorseText you would like to decode. When you're done, press Enter, Ctrl+Z, Enter:
-- _-- / _ . ' _ _ _ / _ . . . . / _ _ _ _ _

Occurrence of dots is: 12
Occurrence of dashes is: 12

The decoded text is: MY NAME IS ANA
Character A occurred 3 time(s)
Character E occurred 1 time(s)
Character I occurred 1 time(s)
Character M occurred 2 time(s)
Character N occurred 2 time(s)
Character S occurred 1 time(s)
Character Y occurred 1 time(s)
nika@NK-MacBook-Air Morse %
```

Here is decoded a personalized input. The output gives the statistics of the input, the decoded text, and the statistics of the output.

If the user wants to decode an existing text, use the proper filename:

```
morse.c -d <filename>
```

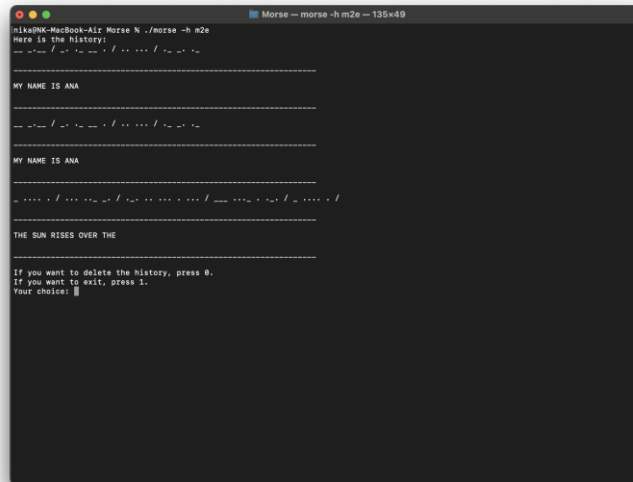


```
-----
If you want to delete the history, press 0.
If you want to exit, press 1.
Your choice: 0
The history was successfully deleted! :3
```

Here is presented the English text to Morse code history of translations. Further, the possibility to either delete the history or exit the program is provided.

For viewing Morse code to English text translations:

morse.c -h m2e



```
nikanik-MacBook-Air Morse N ./morse -h m2e
Here is the history:
-----
MY NAME IS ANA
-----
MY NAME IS ANA
-----
THE SUN RISES OVER THE
-----
If you want to delete the history, press 0.
If you want to exit, press 1.
Your choice: 
```

Here is presented the Morse code to English text history of translations. Further, the possibility to either delete the history or exit the program is provided.

Important Note: This code does not manipulate special characters like ,, ", :, ;, ?, and so on, which may be present in English text. When inserting Morse code, use . and _ . Delimit the letters by a space and the words by a forward slash surrounded by spaces (/) to distinguish between words (space - slash - space).