Lab6MorseCodeLab

Generated by Doxygen 1.8.8

Thu Nov 16 2017 15:56:08

Contents

1 Specification

In this project, we will simulate a telegraph station that can encode messages from text to Morse code and decode Morse code back to text. The encoding is accomplished by looking up a symbol in a table and copying its Morse code into the output string. The decoding is implemented with the help of a binary "decoding" tree. Morse code for each letter represents a path from the root of the tree to some node: a "dot" means go left, and a "dash" means go right. The node at the end of the path contains the symbol corresponding to the code.

2 Analysis

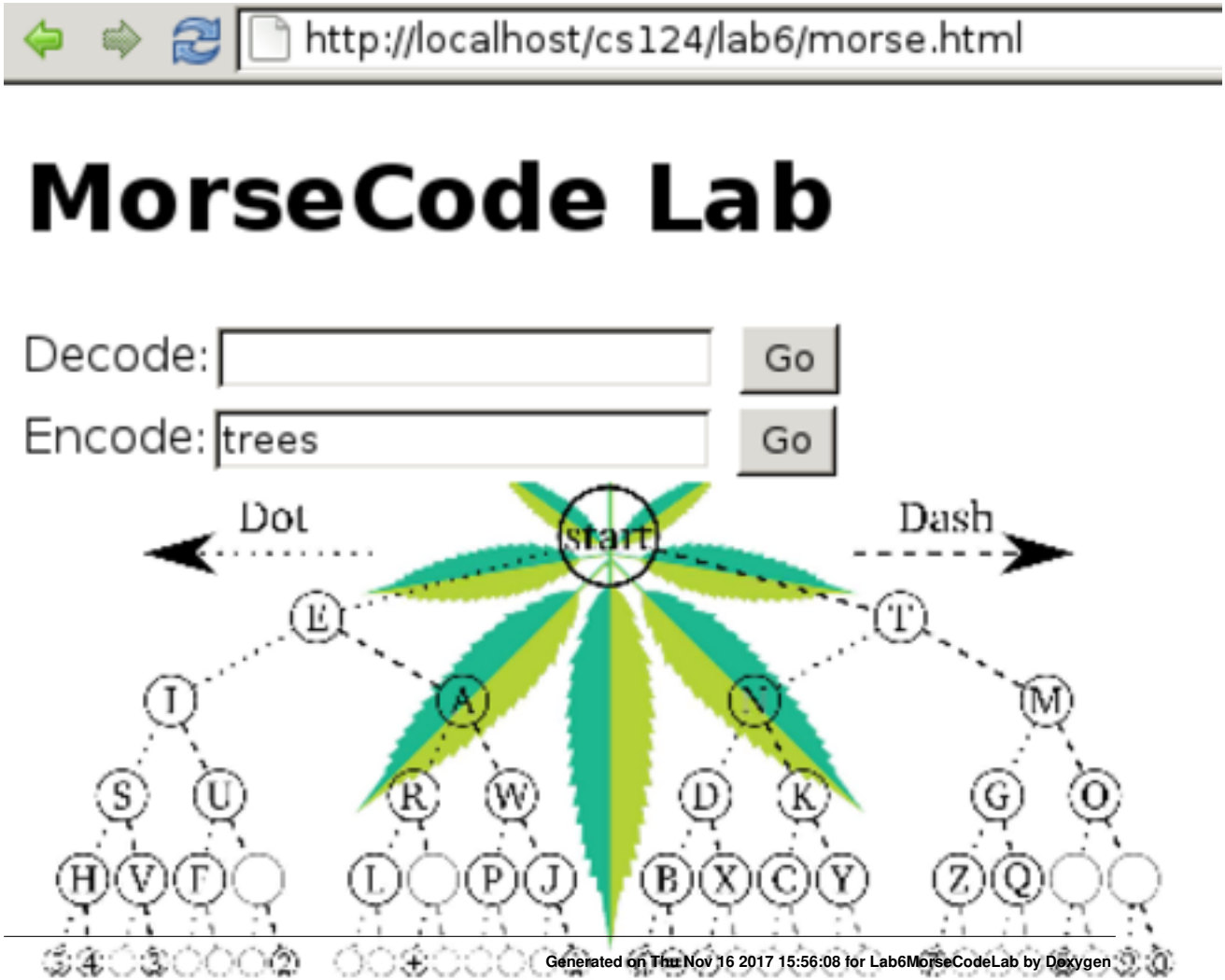
Inputs: Morse code The user entering letters The user entering morse code User selction of encode and decode
Outputs: Morse code to letters letters to morse code

3 Design

The desgin is to have html to display our input and output and have the C++ do all of the work. we have a total of 5 functions. openTree.cpp: Builds the tree needed for the morse code translations.Sets the root node to null and builds from the tree from that function builds from the Morse table structure from the privateclass. encode.cpp: Converts text into morse code. It finds each symbol in the table(skips it if not found). decode.cpp: Converts morse code into text. It follows the path determined from the string parameter and once it reaches the end, it stores the letter in the string named text which is returned at the end of the functions. closeTree.cpp Destroys the tree that was built with the buildTree function. Uses recursion to destroy each node of the tree.

4 Testcase1

This is our home html page. Let's try to encode the word "trees". After we hit the go button, we should be redirected to a new html page with "trees" translated to morse code.

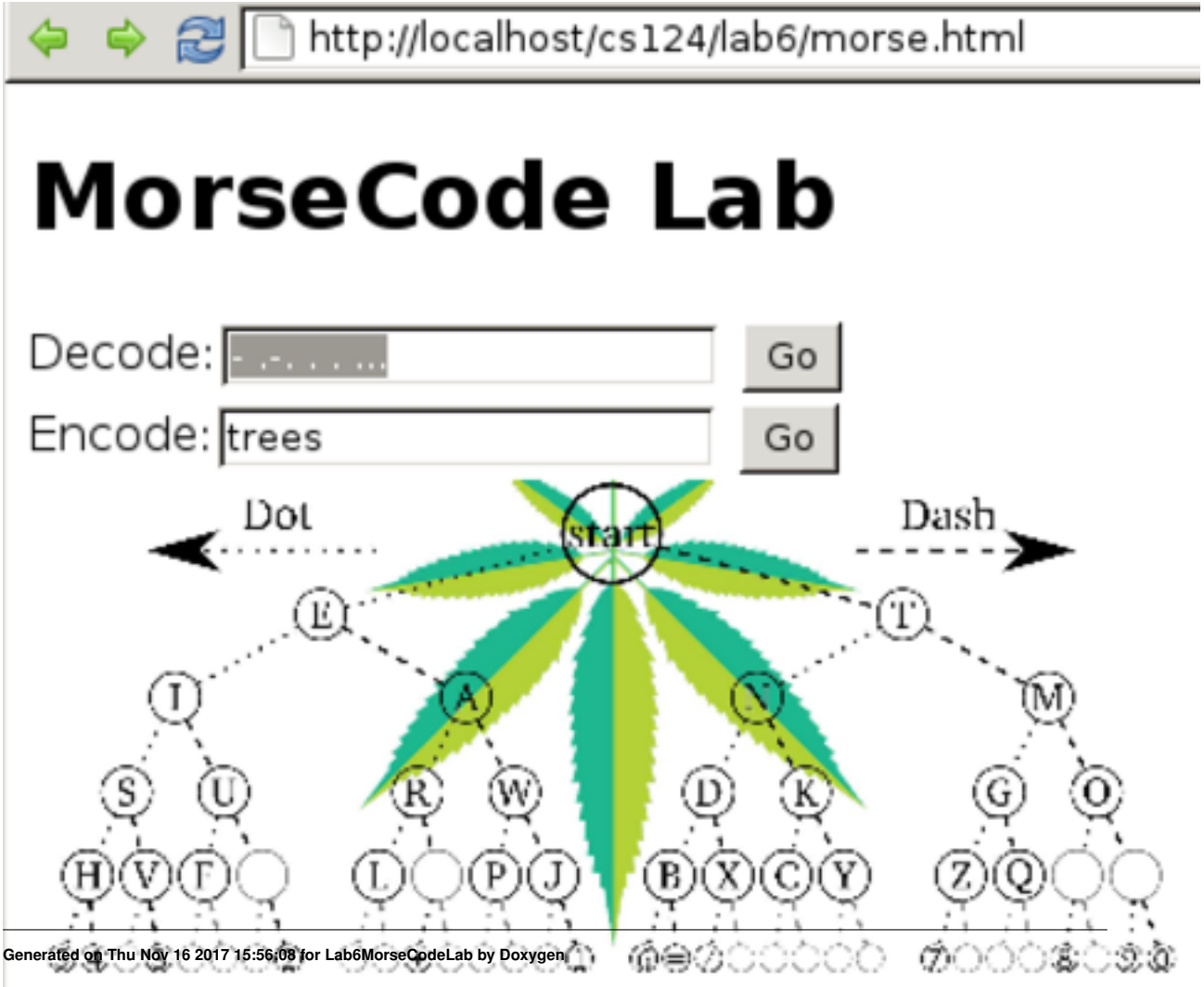

http://localhost/cs124/lab6/morse.html
MorseCode Lab
Decode:
Go
Encode:
trees
Go
Dot
Dash
start
E
T
I
A
N
M
S
U
R
W
D
K
G
O
H
V
F
L
P
J
B
X
C
Y
Z
Q
5
4
3
2
+

5 Testcase2

Nice it worked. Now let's copy the output from our morse code and validate it using our decode function.

6 Testcase3

After copying pasting into our decode text field. We should get an abc translation when we click the go button.

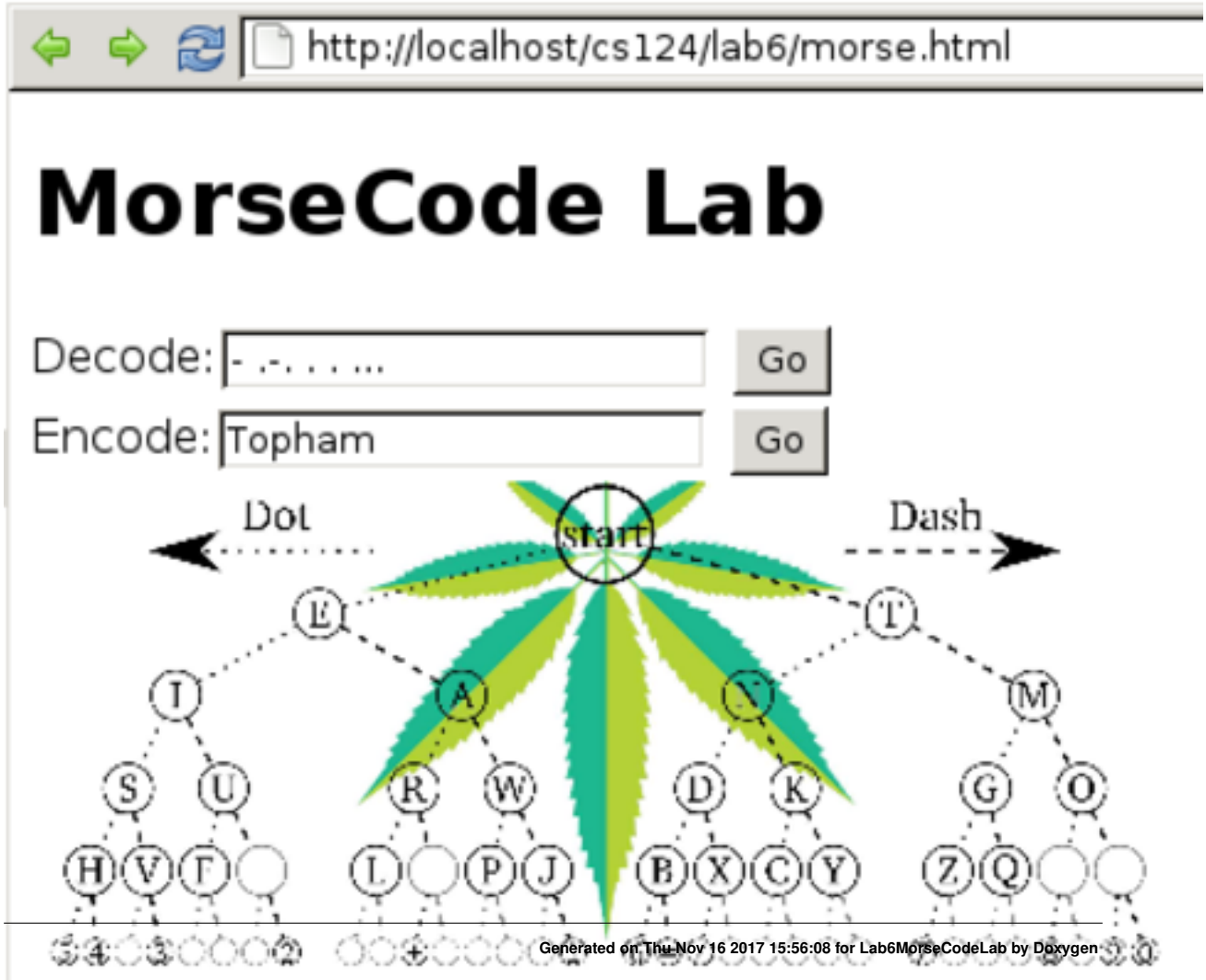

http://localhost/cs124/lab6/morse.html
MorseCode Lab
Decode:
- .-. . . ...
Go
Encode:
trees
Go
Dot
Dash
start
E
T
I
A
N
M
S
U
R
W
D
K
G
O
H
V
F
L
P
J
B
X
C
Y
Z
Q

7 Testcase4

Great! Our encode function works and our decode function work since our inputs match the outputs.

8 Testcase5

Now let's try another test. Let's encode "Topham". We should receive the morsecode translation of it.


http://localhost/cs124/lab6/morse.html
MorseCode Lab
Decode:
- .-. . . ...
Go
Encode:
Topham
Go
Dot
Dash
start
E
T
I
A
N
M
S
U
R
W
D
K
G
O
H
V
F
L
P
J
B
X
C
Y
Z
Q

9 Testcase6

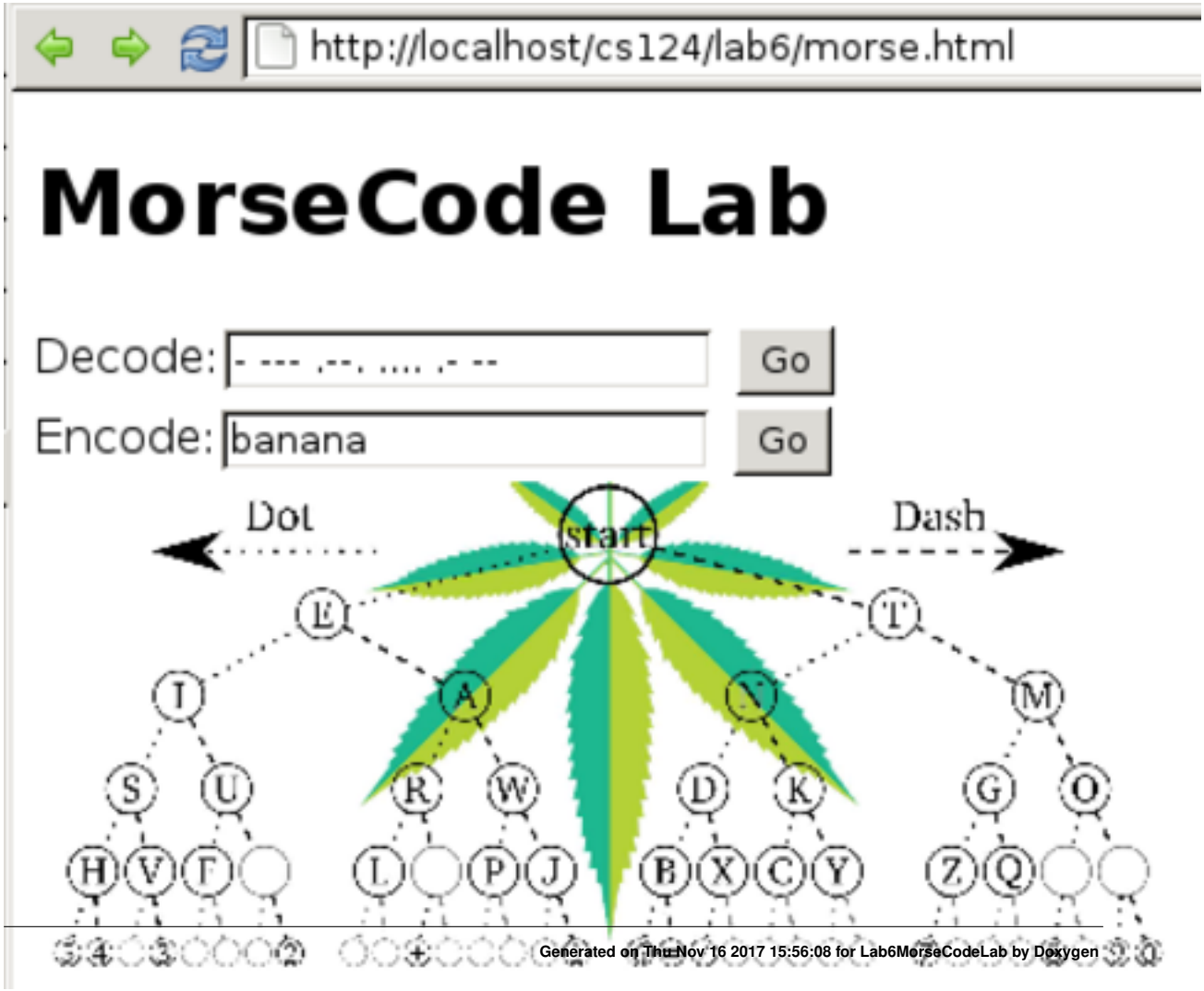
Awesome! let's copy and paste this morsecode in to our decode function to validate it.

10 Testcase7

We should see "TOPHAM" when we submit it.

11 Testcase8

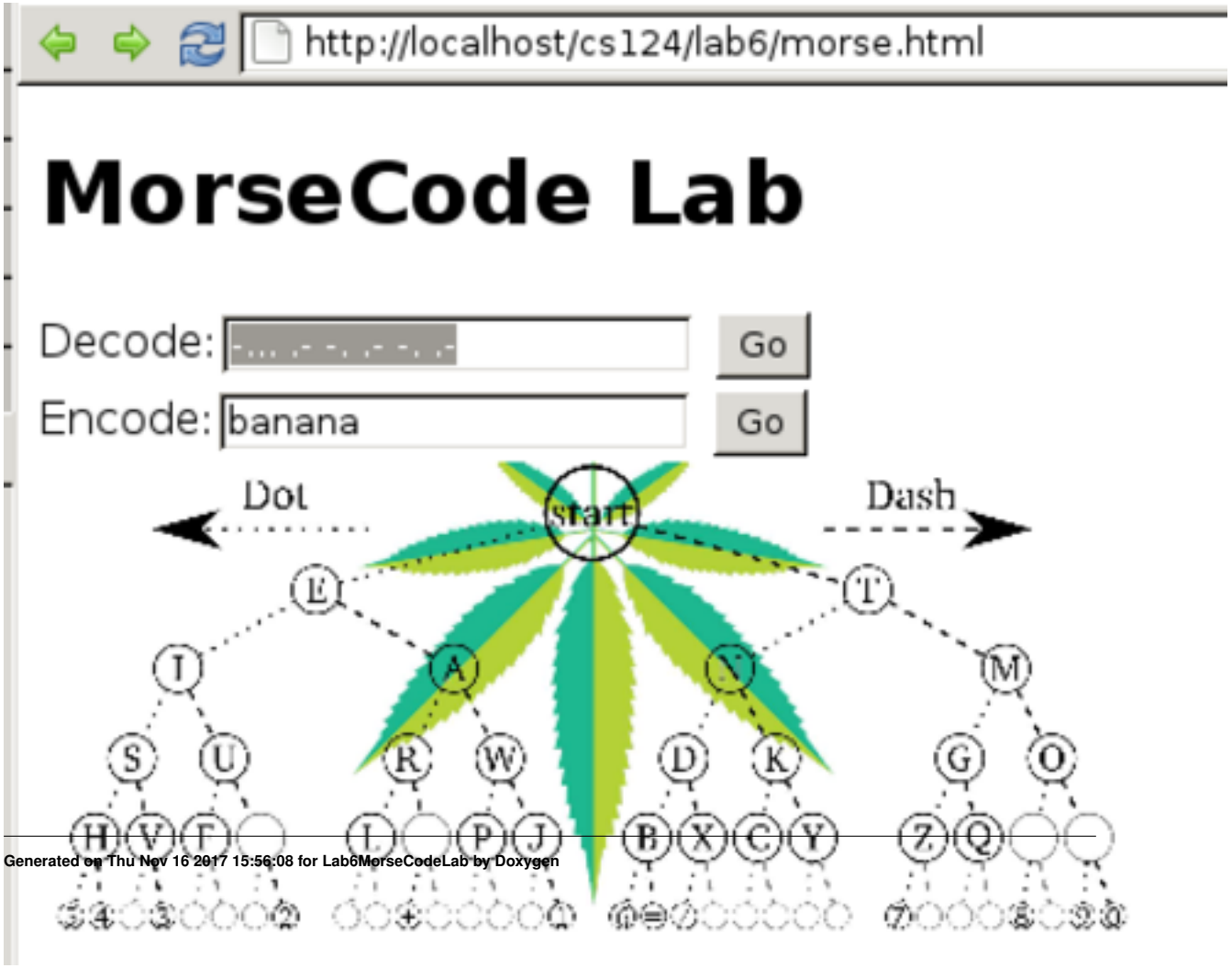
Beautiful. Let's try one more test case to make sure it works.


http://localhost/cs124/lab6/morse.html
MorseCode Lab
Decode:
- --- .--. .... .- --
Go
Encode:
banana
Go
Dot
Dash
start
E
T
I
A
N
M
S
U
R
W
D
K
G
O
H
V
F
L
P
J
B
X
C
Y
Z
Q

12 Testcase9

Here we try our favorite word "banana".

13 Testcase10

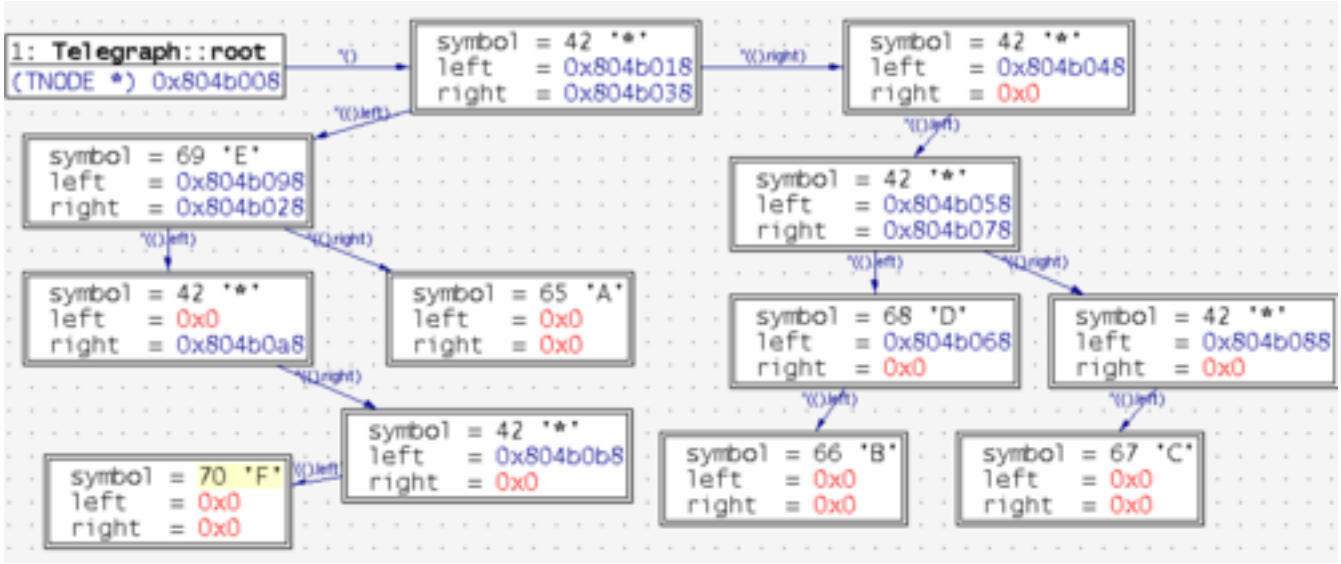


14 Testcase11

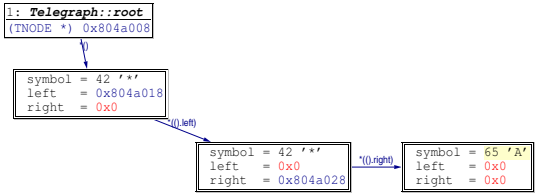
Great test case 3 works too!

15 Testcase12

DDD Tree Example



16 diagram


1: Telegraph::root
(TNODE *) 0x804a008
*()
symbol = 42 '*'
left = 0x804a018
right = 0x0
*(().left)
symbol = 42 '*'
left = 0x0
right = 0x804a028
*(().right)
symbol = 65 'A'
left = 0x0
right = 0x0

17 Class Index

17.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

18 File Index

18.1 File List

Here is a list of all files with brief descriptions:

19 Class Documentation

19.1 Morsecode Struct Reference

```
#include <morse.h>
```

Public Types

- enum { N =7 }

Public Attributes

- char symbol
- char code [N]

19.1.1 Member Enumeration Documentation

19.1.1.1 anonymous enum

Enumerator

N

```
7 {N=7};
```

19.1.2 Member Data Documentation

19.1.2.1 char Morsecode::code[N]

19.1.2.2 char Morsecode::symbol

The documentation for this struct was generated from the following file:

- morse.h

19.2 Telegraph Class Reference

```
#include <morse.h>
```

Public Member Functions

- void Encode (char text[], char morse[])
- void Decode (char morse[], char text[])

Static Public Member Functions

- static void open ()

  open creates the morse code table
- static void close ()
- static void destroyTree ()

19.2.1 Member Function Documentation

19.2.1.1 void Telegraph::close ( ) [static]

```
{
    destroyTree(root);
    root = 0;
}
```

19.2.1.2 void Telegraph::Decode (char *morse[], char text[])

```
{
    char *dd;
    TNODE *node;
    node = root;
    //char *t;
    // For each char in the encoded message (can be
    // a dot, a dash, or a space):
    for (dd = morse; *dd; dd++) {
        if(*dd == '.')
            node = node->left;
        else if(*dd == '-')
            node = node->right;
        else {
            *text++ = node->symbol;
            node = root;
        }
    }
    *text = '\0';
}
```

19.2.1.3 static void Telegraph::destroyTree ( ) [static]

19.2.1.4 void Telegraph::Encode (char *text[ ],* char *morse[ ]*)

```
{
    int i;
    char c, *t, *dd; // t points to text;
    // dd points to a string of dots and dashes.
    for (t = text; *t; t++) {
        c = toupper(*t);
        // If space, add a space to the morse string:
        if (c == ' ') {
            *morse++ = ' ';
            continue;
        }
        // Find this symbol in the MORSECODE table;
        // skip this symbol if not found:
        for (i = 0; table[i].symbol; i++)
           if (table[i].symbol == c) break;
        if (!table[i].symbol) continue;
        // Copy its code into the morse string:
        dd = table[i].code;
        while (*dd) *morse++ = *dd++;
        // Add one space to separate letters:
        *morse++ = ' ';
        }
    *morse = '\0';
}
```

19.2.1.5 void Telegraph::open ( ) [static]

open creates the morse code table

```
{
```

```
    char* dd;
    Telegraph::root = new TNODE;
    TNODE* node; TNODE* nextnode;
    for (int i = 0 ; i < N; i++)
    {
        node = root;
        for(dd = table[i].code; *dd ; dd++)
        {   // loops through each char of code
            if(*dd == '.')
            {
                nextnode = node->left;
                if(not nextnode)
                {
                    nextnode = new TNODE;
                    node->left = nextnode;
                }
            }
            else if(*dd == '-')
            {
                nextnode = node->right;
                if(not nextnode)
                {
                    nextnode = new TNODE;
                    node->right = nextnode;
                }
            }
            else std::cerr << "unknown morse code" << std::endl;
            node = nextnode;
        }   // not dash, not dot, therefore
        // it must be null, so assign symbol
        node->symbol = table[i].symbol;
    }
```

```
}
```

The documentation for this class was generated from the following files:

- morse.h
- morse.cpp

19.3 TNODE Struct Reference

```
#include <morse.h>
```

Collaboration diagram for TNODE:

Public Member Functions

- TNODE ()

Public Attributes

- char symbol
- TNODE * left
- TNODE * right

19.3.1 Constructor & Destructor Documentation

19.3.1.1 TNODE::TNODE ( ) [inline]

```
17              { // Constructor
18          symbol = '*';
19          left = 0;
20          right = 0;
21      }
```

19.3.2 Member Data Documentation

19.3.2.1 TNODE* TNODE::left

19.3.2.2 TNODE* TNODE::right

19.3.2.3 char TNODE::symbol

The documentation for this struct was generated from the following file:

- morse.h

19.4 TREENODE Struct Reference

```
#include <tree.h>
```

Collaboration diagram for TREENODE:

Public Attributes

- SOMETYPE data
- TREENODE * left
- TREENODE * right

19.4.1 Member Data Documentation

19.4.1.1 SOMETYPE TREENODE::data

19.4.1.2 TREENODE* TREENODE::left

19.4.1.3 TREENODE* TREENODE::right

The documentation for this struct was generated from the following file:

- tree.h

20 File Documentation

20.1 lab.dox File Reference

20.2 main.cpp File Reference

```
#include "morse.h"
```

Functions

- int main ()

20.2.1 Function Documentation

20.2.1.1 int main ( )

```
{
    Telegraph::open();
    Telegraph t;
    std::string str = getenv("QUERY_STRING");
    std::string strOG = str; std::string strInput = str;
     strOG.erase(1,30); // extract first char
     strInput.erase(0,2); // extract the string from the text field
     if ( strOG == "d") { // Split string where the +'s are instead of converting to spaces
         for (int i = 0; i < strInput.size(); i++) // Convert all +'s to spaces
             if(strInput[i] == '+')
                 strInput[i] = ' ';
```

```
        strInput += " ..";
        //std::cout << strInput << std::endl;
        char text[strInput.length() + 1];
        strcpy(text, strInput.c_str());
        char eMorse[600];
        t.Decode(text, eMorse);
        std::cout << "<html><h2>ABC:</h2><h3>" << eMorse << "</h3></html>" << std::endl;
    }
    else if ( strOG == "e") {
        char morse[strInput.length() + 1];
        strcpy(morse, strInput.c_str());
        char dText[600];
        t.Encode(morse, dText);
        std::cout << "<html><h2>Morsecode:</h2><h3>" << dText << "</h3></html>" << std::endl;
    }
    Telegraph::close();
}
```

20.3 morse.cpp File Reference

```
#include "morse.h"
```

20.4 morse.dox File Reference

20.5 morse.h File Reference

```
#include <iostream>
#include <cctype>
#include <string.h>
```

Classes

- struct Morsecode
- struct TNODE
- class Telegraph

20.6 morseCode.h File Reference

Variables

- Morsecode table []

20.6.1 Variable Documentation

20.6.1.1 Morsecode table[]

Initial value:

```
= {
  {'A', ".-"},     {'B', "-..."},    {'C', "-.-."},   {'D', "-.."},
```

```
  {'E', "."},       {'F', "..-."},     {'G', "--."},     {'H', "...."},
  {'I', ".."},      {'J', ".---"},     {'K', "-.-"},     {'L', ".-.."},
  {'M', "--"},      {'N', "-."},       {'O', "---"},     {'P', ".--."},
  {'Q', "--.-"},    {'R', ".-."},      {'S', "..."},     {'T', "-"},
  {'U', "..-"},     {'V', "...-"},     {'W', ".--"},     {'X', "-..-"},
  {'Y', "-.--"},    {'Z', "--.."},
  {'0', "-----"},   {'1', ".----"},    {'2', "..---"},   {'3', "...--"},
  {'4', "....-"},   {'5', "....."},    {'6', "-...."},   {'7', "--..."},
  {'8', "---.."},   {'9', "----."},
  {'.', ".-.-.-"},  {',', "--..--"},   {'?', "..--.."},
  {'\0', ""}
}
```

20.7 tree.h File Reference

Classes

- struct TREENODE

Functions

- void Destroy (TREENODE *root)
- int Compare (const SOMETYPE data1, const SOMETYPE data2)
- void Copy (const SOMETYPE data1, SOMETYPE data2)

20.7.1 Function Documentation

20.7.1.1 int Compare (const SOMETYPE *data1*, const SOMETYPE *data2*)

20.7.1.2 void Copy (const SOMETYPE *data1,* SOMETYPE *data2*)

20.7.1.3 void Destroy (TREENODE * *root*)

Index