**The Message:**

If you can read this the code is working well done and party one

I[ppis1]IN[ii]YOU[ppy]CAN[vm]READ[vvi]THIS[dd1]THE[at]CODE[nn1]IS[vbz]WORKING[vvg]WELL[rr]DONE[vdn]AND[cc]PARTY[nn1]ON[ii]

**Explanation:**

To start a message is sent to a decode method.

The decode method runs through the length of the message (string) and checks for each possible word that could be in each segment of the message (Example: check for words that could come from message[0] to message[1], message[0] to message[2], and so on)

For each word that it finds it adds the word to a possibilities list along with the length of morse code that was used and the score of the word (which to start the score is just 1)

After this a decode recursive function is called sending in the original message

The decoded recursion runs through the possibilities list, for each item in the list it runs through a helper decoded function, which takes the word and the rest of the message not used along with the length of the morse code used from the last word and the last frequency between the words before it.

This function runs through the rest of the message and finds possible words that can be added onto the last word to make a sentence, it does this by checking if there are any possible words, getting all the possible words, running through each possible word and making sure that it is a valid word and meets the frequency threshholds.

If it is a possible word it will get added to a list (tagged on with the last word so as to not lose track of what the sentence is) and called the helper decode function passing in the word that was just found, the rest of the message, the length of morse code used, and the current score of the possible sentence.

This will continue until there is no more message left to check, once there is no more message left to check the function will return back and start from the last time the helper decode function was called (most likely this will be in a recursive call as well), it will continue to check, add to the list, and call itself as long as there are possibilities found.

Once all the possibilities are found I do some extra things such as get rid of any extra garbage found in my list (if there is any) and quicksort my list to find the best possible options of translation.

I then return this final list of best possible options.

**Help Received:**

* Drew Hurdle helped me understand the logic behind recursion a little more since I was very fuzzy on the subject, not having used it in quite a while.

**Mathematical Analysis:**

After doing some tests, I found that the largest upperbound I could get through many morse code strings was 35. So I decided to use 35 as my upperbound as it seems to be my worst case.

T(n)=35^n\*T(n-1)

T(n)=35^n\*35^(n-1)\*T(n-2)

T(n)=35^n\*35^(n-1)\*35^(n-2)\*T(n-3)

T(n)=35^n!\*T(n-k)

O(n!)

**Empirical Analysis:**

All tests are run at a 170 Frequency Threshold.

I decided to use message length, English message, the rank the message was placed, the possible words that had been looked at, and the completion time because I felt these were vital to understanding how the algorithm works. The message length shows how much of a message the algorithm has to consider to find possible words, while the final English message has to match with what the algorithm comes up with. The rank placed shows how well the algorithm was able to achieve the solution state using the possibilities that were run through, each possibility run through includes every word found that could match a segment of the string of morse code given. These possibilities could have disregarded before any frequency thresholds were run if they contained garbage characters, or they could have been disregarded if they did not meet the frequency requirements, finally they could have been included as a possibility of the solution if it meets both requirements suggested. The completion time shows how long it took for the algorithm to go through each one of the steps I have mentioned and find the solution.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Message Length | English Message | Rank Placed | Possibilities run through | Completion Time |
| 30 | To Be Or Not To Be | 1 | 6102 | 5 milliseconds |
| 77 | Life Is Like A Box Of Chocolates | 11 | 41895 | 18 milliseconds |
| 132 | If You Can Read This The Cod Is Working Well Done And Party One | 10 | 3068988 | 569 milliseconds |