**Language Recognition**

By Havryliuk Danylo

Student of the first year of the

Charles University, MFF

Computer Science

**Language Recognition Program Programming Documentation**

**Initialization:**

When the program starts, it initializes two constants, definitelyCorrect and definitelyNotPossible, with predefined values (-1 and -2, respectively). This is very useful, because then each language will receive a score - the relative probability that this particular language is the language of the text, and these two constants will be used to determine the scores. The score will always be positive, so making them negative is a good idea.

**Main Method (User Interaction):**

The Main method is the start point for the program.

It starts by displaying a welcome message and then enters a continuous loop to accept user input. Also, there is input correctness check, by just checking if there are any extra symbols/numbers, that break the text.

Then the program checks if the input is not empty or just whitespace. If it is, it prompts the user to enter valid text.

**Language Recognition (RecognizeLanguage Method):**

The core language recognition process is handled by the RecognizeLanguage method.

This method takes the inputText as its parameter and is responsible for determining the most likely language.

It defines and initializes various language-specific dictionaries and parameters for supported languages, including letters, letter combinations (digraphs and trigraphs), characters, and popular words.

It also initializes a languageScores dictionary to store scores for each language**.**

Then it starts CalculateScore Method for each language.

**Calculate Score (CalculateScore Method):**

In CalculateScore method, there is a loop that iterates through each character in the inputText.

For each character, it performs the following checks:

If some of the characters is not in language alphabet it returns score of this language equal to DefinitelyNotPossible, because we know, that it can’t be this language.

Then, again loop through each character in inputText, that checks if the character is specific for this particular language (it exists only in this language alphabet). If yes, so we found language, and score equal to definitelyCorrect is returning.

It also checks if there some popular word from this language in text, if it finds it - return definitelyCorrect. (again, using Regex, but now with @"\b" + … + @"\b", because we want to find our popular word in full word in text, not just in part of word).

Then, if program didn’t find the result in simple way, it goes further. It starting to calculate score for language. For this, program has 2 dictionaries with science confirmed results: first has most popular 2-3 combinations of letters, second – less popular, but also often used. Then, using Regex (**regular expression)**, it finds the number of repetitions of all combinations. Score would be equal to number of popular combinations plus (number of less popular combinations/2). Thus, despite that the percentages are not directly calculated, we have a relative probability expressed by the value of the particular language score / the sum of all scores.

**Analyze Scores (AnalyzeScores Method):**

The AnalyzeScores method analyzes the language scores to determine the detected language.

It creates two lists – definitelyCorrectLanguages ​​and possibleLanguages, fills them.

If the list of definitelyCorrectLanguages ​​is empty - takes the language with the maximum score from the list of possibleLanguages.

If, for example, there are 2 or more languages ​​in definitelyCorrectLanguages, it means that the text is written in more than one language (for example there are letters from and Ukrainian and English) , and it gives this in input.It returns the detected language or relevant messages as if there's insufficient data, etc.

**IsLetterValidForLanguage Method:**

This method checks if a given character is specific to a particular language. It takes a character, language, and language-specific character set as parameters. It returns true if the character is valid for the language and false otherwise. It is used in CalculateScore to check if character specific for language.

**Additional**

The program interacts well with incorrect input data, quite accurately gives answers in non-standard situations (the text is not written in a supported language / the text is in several languages). Also, was added support for text symbols such as ?!,. to be able to insert texts. Also added ¿ for Spanish.

Thank you, for reading this. Have a good day and …

**Enjoy using the Language Recognition Program!**