Translator Project

Members: Nathan Darroch, Ryan Hutton, Douglas Parkinson, Momoko Werner

Analysis of Problem:

The project is to develop a piece of software able to perform language translation to/from English to French. At a minimum the program should be able to translate a sentence word for word giving a literal translation. This program should take in a user input for what they want to do in the program and offer choices for adding words, deleting words, displaying the dictionary files and also translating to and from French. When the user translates a sentence the program should output the translated sentence with the time it took to translate it.

(Additional research for this problem can be found in the folder “Additional Research”)

Requirements:

Functional:

1. The program shall provide dictionaries to translate from one or more language to English and from English to French.
2. The program shall be able to translate both phrases and individual words
3. The program shall translate a text from one language into another, and output the time taken to complete the translation.
4. The program shall provide an option to enable users to add new words to the translation dictionary when your translator cannot find the word in the dictionary.
5. The program shall provide an option to remove a word(s) from the dictionary.
6. The program shall provide an option to display the dictionaries.
7. The program may have a GUI to display the input text and the translation as well as graphical input options for the options other than the translation.
8. The program may translate groups of words and phrases as well as individual words.
9. The program may support translation from additional languages.

Non-Functional:

1. The translator shall require the java platform to be installed on the device running the program.
2. The translator shall not have a process runtime of over a minute for each translation.
3. The translator shall always have the dictionary files in its project directory and be able to access them.

Functional Requirements Completed:

1 Completed

2 Completed

3 Completed but without displaying the time taken for the translation

4 Not Completed

5 Completed

6 Completed

7 Unknown

8 Completed

9 Not Completed

Classes:

Candidate Classes:

|  |  |  |
| --- | --- | --- |
| **Candidate Classes** | **Accept / Reject** | **Reason** |
| Main | Reject | Too vague. |
| Translator | Accept | Specific to the translation taking place. |
| Menu | Accept | Name refers the input handling from the user in the menu system. |
| BinaryTree | Reject | Too vague, doesn’t specify the data being held within it. |
| TreeNode | Reject | Too vague, doesn’t tell you what the data in the node is. |
| DictionaryTree | Accept | Refers to the dictionary of words which this binary tree will be storing. |
| WordNode | Accept | Refers to the word string data contains within each node. |

Class Descriptions, fields and methods:

**Menu Class:**

|  |  |
| --- | --- |
| **Method or Field Name:** | **Description:** |
| main() | The main method for the program to start at which is run externally from the java virtual machine when the program starts. |
| displayMenu() | The menu method for displaying options to the user. |
| processUserChoices() | Method used for handling input and choices from the user and run the corresponding method. |
| translate() | Method to run the translation option if the user selects it. |
| addWord() | Method for running the method for adding a word to the dictionary if it is not in there if the user selects that option. |
| delWord() | Method for running the method to deleting a word from the dictionary if the user selects that option. |
| display() | Method for running the method to display the contents of the dictionary if the user selects that option. |
| Translator translater | An instance of the Translator class which is used to call non static methods within the main methods scope. |

**Translator Class:**

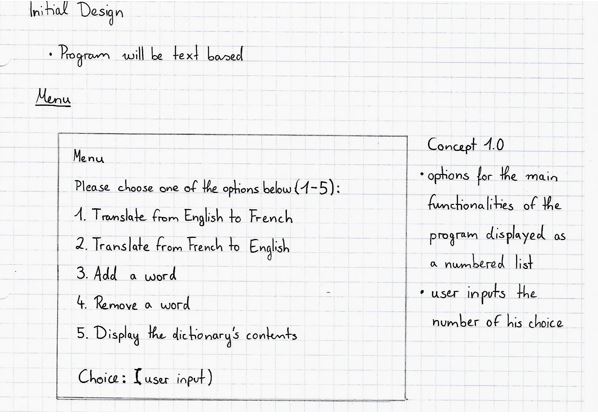
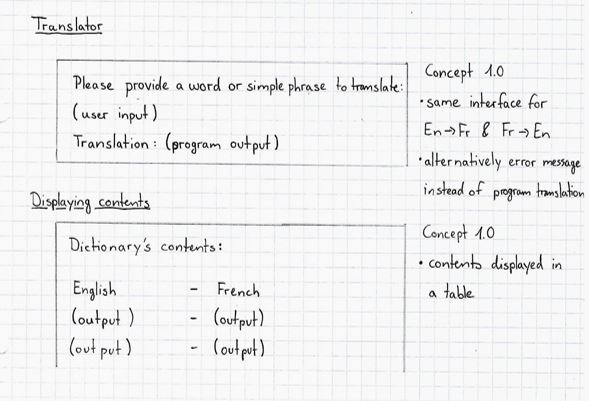
|  |  |
| --- | --- |
| **Method or Field Name:** | **Description:** |
| translateWord() | Takes in a word and searches for it within the binary tree. Once it has been found it returns it’s translated equivalent. |
| translateSentence() | Takes in a sentence and breaks it down into its words then calls translateWord() to translate each word to translate the whole sentence. |
| addWords() | Writes a given word to an external file with its French translation. |
| deleteWords() | Takes in the word you want to delete and removes it from an external file. |
| overrideCustomDictionary() | Writes new news words into custom dictionary files. |
| openUserFileAndRead() | Opens a file to read it. |
| readFile() | Reads in lines from file into one string and returns it. |
| createCustomDictionaryFiles() | Creates custom dictionary files if they are not already there. |
| loadDictionary() | Reads the external dictionary files and loads them into the DictionaryTree data structure. |
| fileExistsAndCanRead() | Checks if an external file is accessible. |
| displayDictionary() | Calls the display tree method in the DictionaryTree to show all the words in alphabetical order. |
| DictionaryTree english | An instance of the DictionaryTree class which is used to store all the words and their translated counterparts. Is ordered by the english words. |
| DictionaryTree french | An instance of the DictionaryTree class which is used to store all the words and their translated counterparts. Is ordered by the french word. |

**DictionaryTree Class:**

|  |  |
| --- | --- |
| WordNode root | Used to hold the reference for the WordNode root of the binary tree data structure. |
| addToTree() | A method used to add a word to the binary tree data structure. |
| searchTree() | A method to search the binary tree data structure for a specific word and return it. |
| displayTree() | A method to print out all the words in the binary tree structure. |
| traverseTree() | Goes through each node in the tree and prints it out in order. |
| removeFromTree() | A method to remove a word from the binary tree structure. |
| getRoot() | Returns the WordNode at the root of the tree. |
| setRoot() | Sets the root of the tree to a specific WordNode. |
| isTreeEmpty() | Returns whether or not the DictionaryTree is empty or not. |

**WordNode Class:**

|  |  |
| --- | --- |
| String ID | The word in the language we want the WordNode to be ordered by. |
| String english | String for holding the english translation of the word. |
| String french | String for holding the english version of the word. |
| WordNode prev | Used for holding the reference to node previous to this one in the structure. |
| WordNode left | Used for holding the reference to node left of this one in the structure. |
| WordNode right | Used for holding the reference to node right of this one in the structure. |
| getID() | Returns the ID string of the WordNode. |
| getEnglish() | A method that returns the value of english. |
| getFrench() | A method that returns the value of french. |
| getPrev() | A method that returns the reference of prev. |
| setPrev() | A method that sets the reference prev. |
| getLeft() | A method that returns the reference of left. |
| setLeft() | A method that sets the reference left. |
| getRight() | A method that returns the reference of right. |
| setRight() | A method that sets the reference right. |

Interface Designs:

Use Cases:

|  |  |  |  |
| --- | --- | --- | --- |
| Translating | | | **Alternatives** |
| **1** | System | Displays menu |  |
| **2** | User | Chooses to translate (in either direction) |  |
| **3** | System | Prompts user to give a word/phrase to be translated |  |
| **4** | User | Gives input word |  |
| **5** | System | Searches database for translation |  |
| **6** | System | Finds translation | A1 |
| **7** | System | Gives translation as output |  |
| **8** | System | Returns to the menu |  |

|  |  |  |  |
| --- | --- | --- | --- |
| A1 Alternative - Translating | | | **Alternatives** |
| **1** | System | Does not find translation, displays error message |  |
| **2** | System | Returns to the menu |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Adding a word | | | **Alternatives** |
| **1** | System | Displays menu |  |
| **2** | User | Chooses to add a word |  |
| **3** | System | Checks whether word is already contained in the external file |  |
| **4** | System | Adds word to the external file | A1 |
| **5** | System | Returns to the menu |  |

|  |  |  |  |
| --- | --- | --- | --- |
| A1 Alternative – Adding a word | | | **Alternatives** |
| **1** | System | Finds word, displays error message |  |
| **2** | System | Returns to the menu |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Removing a word | | | **Alternatives** |
| **1** | System | Displays menu |  |
| **2** | User | Chooses to remove a word |  |
| **3** | System | Checks whether word is already contained in the external blacklist file |  |
| **4** | System | Adds word to the blacklist file | A1 |
| **5** | System | Returns to the menu |  |

|  |  |  |  |
| --- | --- | --- | --- |
| A1 Alternative – Removing a word | | | **Alternatives** |
| **1** | System | Finds word, displays error message |  |
| **2** | System | Returns to the menu |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Displaying contents | | | **Alternatives** |
| **1** | System | Displays menu |  |
| **2** | User | Chooses to see the entire contents of the dictionary |  |
| **3** | System | Prints the dictionary’s contents | A1 |
| **4** | System | Returns to the menu |  |

|  |  |  |  |
| --- | --- | --- | --- |
| A1 Alternative – Removing a word | | | **Alternatives** |
| **1** | System | Dictionary is empty, displays error message |  |
| **2** | System | Returns to the menu |  |

Class Diagram:

Menu Class

Translator Class

DictionaryTree Class

WordNode Class

Pseudocode:

Translator Class:

translateWord(String WORD, Boolean TRANSTYPE):

GET WORD

CALL METHOD searchTree() IN DictionaryTree CLASS AND PASS IN WORD

SET tempWord TO THE RETURNED WordNode

IF TRANSTYPE == 1 THEN:

RETURN tempWord.getFrench()

END IF

ELSE DO:

RETURN tempWord.getEnglish()

END ELSE

translateSentence(String SENTENCE):

GET SENTENCE

SET words TO SPLIT OF SENTENCE USING “ ” AS SPLIT SYMBOL

SET translation TO “”

FOR i IN LENGTH OF words DO:

CALL METHOD translateWord() AND PASS IN words[i]

SET translation TO ITSELF + THE RETURNED STRING FROM THE METHOD

END FOR LOOP

RETURN translation

Task Allocation:

Nathan: addWord(), delWord(), translateWord() (Translator class), looking into GUI

Douglas: loadDictionary() (Translator class), data base management and planning (documents for coding and meetings),

Ryan: WordNode class, translateSentence() (Translator class)

Momoko: Menu class, DictionaryTree class, data base management, record management looking into dealing with complex sentences

Evaluation of Project:

During the project we faced many problems for example how we were going to translate multiple words in french to their singular meaning in english. We overcame that problem by searching to see if the two or three words in front of each word we were translating were translatable as one string together instead of each word being translated separately. We worked quite well as a team to solve these by having quite regular meetings and also keeping in contact on line for when we encountered them. Another issue we faced was the dictionary not loading into the binary tree very quickly which we found out was because of how we were inputting the data. We solved this by writing a program that randomly ordered the dictionary files we were using and put them into the same file. This solved the loading issue by taking it’s loading time down from 8 minutes to around 15 seconds. One problem we couldn’t solve in the end was how to keep grammar rules intact while translating as we could find a way to automatically find out which words were verbs, nouns, etc. This made it very hard to code a grammatical check method to make sure the grammar of the english sentence was being passed to the french translation. However we were able to hard code some of the verbs into the french to english translation in order to make the multiple french verb meanings that have connotations with gender and tense simplify to the single english meaning.