Section 1

The main objective of my program is to return rhymes based on any word that has been input after running the program, but the program will return rhymes according to the last two letters of a word that has been input. If there is only one word, numeric or no word insert then the program will find it as an error and return 'No rhyme words found for '. If the word inserted is two or more than two words then the program will go through all the steps from the algorithm to confirm that the word is a letter or not. If it is a word consisting of letters then it will look for the last two letters, afterwards, the program will look for similar last two words from the 'HTTP' link I have given from the Mid-term coursework pdf, which I provided in the program for returning rhymes. The program will return all the similar last two letter words it finds then return as a rhymes.

Section 2

The program has four main functions.

i) bubble sort: The bubble sort algorithm determines if a swap is needed for words, if it does then it goes through the process of swapping alphabetically.

ii) question: This function takes a prompt after running the program, after the user types a word this function gets executed.

iii) getRhymeNotes: In this function, it takes on the prompt that the user has provided and based on the word the function either fills the empty string with rhymes or returns an error.

iv) findRhymes: findRhymes function contains all previous functions, as this is the function which returns all the output values. In addition to that, there are also some additional changes made within this function so, that whenever a word has been typed whether there is a question mark or capital letters the user should get their Rhymes output.

Section 3

Pseudocode:

```
Import readline
Import https

Readline.createInterface ({
    Input: process.stdin,
    Output: process.stdout
})

Let dictionary = []

Function Bubblesort(vector)
    For $1 \leq j \leq n - 1$ do
        Count ← 0
        For $1 \leq j \leq n - 1$ do
            If vector[j+1] < vector[j]
                Swap(vector, j, j + 1)
                Count ← count + 1
            End if
        End for
        If count = 0 then
```

```
                Break
            End if
        End for
        Return vector
End function

Function question(prompt)
   return a new Promise with (resolve)
        Call read_line.question with (prompt, resolve)
End Function

Function getRhymeNotes()
   return a new Promise with (resolve, reject)
      Initialize note as an empty string
      Call https.get with (‘http://link’, rhyme_notes)
            On 'data' event of rhyme_notes
               Append fill to note
             End On
            On 'end' event of rhyme_notes
               Resolve the Promise with note split by '\n'
            End On
            On 'error' event of rhyme_notes
               Reject the Promise
            End On
 End Function

Async Function findRhymes()
   dictionary ← await getRhymeNotes()
   input ← await question('Please enter a word: ')
   word_input ← convert input to lowercase and question mark
```

```
    last_two_lettersInput ← get the last two letters from the
                            word_input
    rhymes ← filter dictionary to find words that end with the same
             two letters as word_input and are not equal to
             word_input
    If rhymes.length > 0 then
        Print "Rhyme with last 2 words from 'word_input': " and
        the sorted rhymes
    Else
        Print "No rhyme words found for 'word_input'"
    End If
    Close read_line
End Function

Call findRhymes()
```

Section 4

List of Data structures:

1. **bubblesort algorithm**: bubble sort algorithm repeatedly steps through the list, compares nearest elements and swaps them if they are in the wrong order. This process is repeated until the list is sorted.
2. **Promise:** Promises are used in the question and getRhymeNotes functions. As I have decided to use the async method, promise is required for the completion or failure of an asynchronous operation.
3. **Filter algorithm:** This is used in the findRhymes function. The filter() method creates a new array with all elements that pass the test implemented by the provided function.

4. **async/await:** This is also used in the findRhymes function. The async function is necessary to use for using await function. The await operator is used to pause the execution of the function until the Promise resolves. If the promise is resolved then the code gets executed.
5. **Array Data structure**: Arrays are used throughout the code to store and manipulate collections of data.

Section 5

Video link: https://youtu.be/_DhcP6wYpfM?si=cobHtfeXhJZ0OiuV

Section 6

The implementation of the program does not have any defects.