CSE 102 Assigment IX

In this project, I implemented a program that processes question-answer pairs from a text file named database.txt to generate character-based one-hot vectors. The program is console-based and uses a series of functions to parse the input data, build a unique character set, and generate the corresponding encodings.

At the start of the program, the parsing() function is called to read the question-answer pairs from the database.txt file. This file is structured such that each question begins with a Q: prefix and each answer with an A: prefix. The function removes these prefixes and stores the remaining text in separate questions and answers arrays. It continues reading until it encounters the delimiter ---, which separates each question-answer pair. The number of pairs is counted and stored in the pairs_counter variable.

Once the sentences are loaded, the program extracts the unique characters present in the text. This is handled by the tokenizeSentence() and buildCharset() functions. tokenizeSentence() splits each sentence into words based on spaces and passes these words to the buildCharset() function. The buildCharset() function iterates over each character in a word and adds it to a charset array if it is not already present. This ensures that the charset contains only unique characters. After processing all sentences, a single space character is manually added to the charset to account for the spaces lost during tokenization.

Next, the program calculates the maximum possible encoding length for each sentence using the findMaxEncodingLength() function. This function identifies the longest sentence among all questions and answers and multiplies this length by the number of unique characters in the charset. This maximum length ensures that each sentence can be fully encoded without truncation.

With the character set and maximum encoding length determined, the program then generates the one-hot vectors using the encodeSentence() function. This function left-pads each sentence with zeros to match the maximum encoding length, then iterates through each character, setting the corresponding position in the encoding array to 1 based on the character's index in the charset.

Finally, the program writes the generated data to two separate files. The writeCharsetToFile() function saves the charset to a file named charset.txt, ensuring that the exact set of unique characters is preserved. The writeToFile() function outputs the one-hot encoded vectors to a file named embeddings.txt, including metadata such as the maximum encoding length, embedding dimension, and total number of pairs.

- **Metadata Part**

```
Metadata:
Max encoding length: 96
Embedding dimension: 66
Total pairs: 141
#####

Question: 00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
Answer: 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
---
Question: 00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
Answer: 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
---
Question: 00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
Answer: 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
---
Question: 00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
Answer: 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
---
Question: 00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
Answer: 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
---
Question: 00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
Answer: 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
---
Question: 00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
Answer: 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
---
Question: 00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
Answer: 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
---
```