Assignment 3 By: Dinesh, Junaid, and Shahzan

The primary data structure we used is a Trie. This is a special data structure that stores strings with the properties of a tree. Our code consists of the RWayTrieCode class which implements the following: Insert, Search, StartsWith, and FindWords.

RWayTrieCode Class:
Description: Represents a node in R-Way Trie, containing an array of child nodes and a boolean flag indicating the end of a word.
Constructor: Initializes a new node with an array of 126 child nodes (RWayTrieNode[126]) and sets the end-of-word flag to false.
Time Complexity: O(1) for initialization.

Constructor:

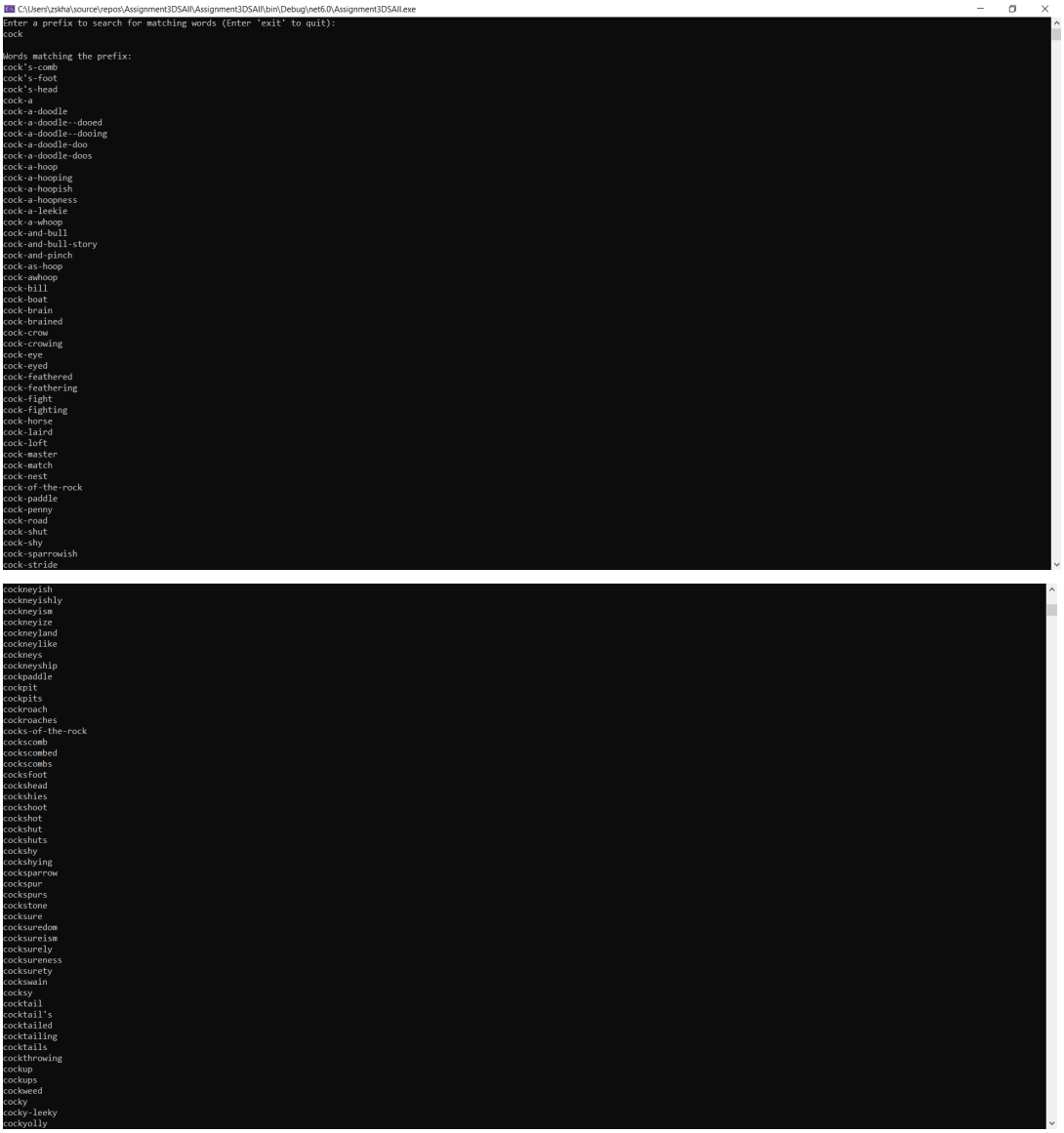
Methods for RWayTrie:

1. **Insert:** This method inserts a word into the trie
   Input: string word, string value
   Operation: Iterates through the trie and creates nodes to accommodate the insertion of the word
   Time Complexity:O(n)

2. **Search(string word):** Searches for a word in the trie
   Input: string word
   Operation: Traverses the trie to find the input word, returning the associated value if found.
   Time Complexity: O(n)

3. **StartsWith:(string prefix):** Find all words in a trie with a given prefix
   Input: string prefix
   Operation: Traverses the trie to find all words that start with the given prefix.
   Time Complexity: O(p + m) for prefix match
   .

Testing:

Case 1:
Input: Cock Expected Output: A lot of Words Actual Output: 

Case 2:

Input: Ui

Expected Output: 8 words

Actual Output:

```
Enter a prefix to search for matching words (Enter 'exit' to quit):
ui

Words matching the prefix:
uily
uinal
uintahite
uintaite
uintaites
uintathere
uintjie
uitspan
```

Case 3:

Input: Hw

Expected Output: 6 words

Actual Output: