**Project Boggle**

**Problem Statement:**

Boggle is a word game designed by Allan Turoff and distributed by Hasbro. It involves a board made up of 16 cubic dice, where each die has a letter printed on each of its 6 sides. At the beginning of the game, the 16 dice are shaken and randomly distributed into a 4-by-4 tray, with only the top sides of the dice visible. The players compete to accumulate points by building *valid* words from the dice, according to these rules:

* A valid word must be composed by following a sequence of *adjacent dice*—two dice are adjacent if they are horizontal, vertical, or diagonal neighbors.
* A valid word can use each die at most once.
* A valid word must contain at least 3 letters.
* A valid word must be in the dictionary (which typically does not contain proper nouns).

**Steps to do:**

1. Read the given dictionary of words into TrieSET so as to save the space and time for the retrieval of a particular word.
2. Recursively find all the possible words by taking one character a time by taking adjacent dice.
3. Insert the valid words into Set for storing all valid words from the word for space and time constraints

**Related Concepts:**

* Two dimensional boolean array
* TrieSET
* Set
* Programming language - java

**Code:**

**public class BoggleSolver**

**{**

**// Initializes the data structure using the given array of strings as the dictionary.**

**// (You can assume each word in the dictionary contains only the uppercase letters A through Z.)**

**//time complexity O(n)**

**//space complexity O(n)**

**public BoggleSolver (String[] dictionary)**

**// Returns the set of all valid words in the given Boggle board, as an Iterable.**

**//time complexity O ()**

**//space complexity O ()**

**public Iterable<String> getAllValidWords (BoggleBoard board)**

**// Returns the score of the given word if it is in the dictionary, zero otherwise.**

**// (You can assume the word contains only the uppercase letters A through Z.)**

**//time complexity O (1)**

**//space complexity O (1)**

**public int scoreOf (String word)**

**}**

**Difficulties Faced:**

* TrieST didn’t work as the space allocation is more when compared to project requirements.

**Test Cases:**

* check scoreOf() on various dictionaries.

dictionary = dictionary-nursery.txt

\* dictionary = dictionary-yawl.txt

- word = 'REACHING'

- student score = 0

- reference score = 11

- failed on trial 1 of 20000

* memory with dictionary-algs4.txt (must be <= 2x reference solution)

\* memory of dictionary[] = 450264 bytes

\* memory of student BoggleSolver = 36239448 bytes

\* memory of reference BoggleSolver = 5091200 bytes

\* student / reference = 7.12