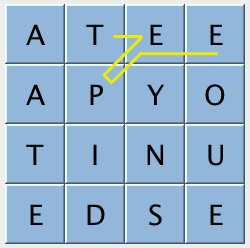
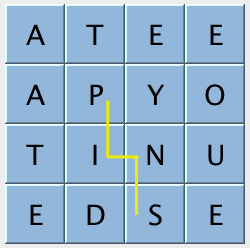
**Boggle Solver**

**Problem Statement** : This projects aims to find all the possible words when a 4 X 4 array of characters is given to the user . If the user picks a valid word , he will be awarded a score based on the length of that valid word .



Some of the data structures and concepts that I have used in this project are :

1.TrieSET

2.HashSET

3.Arrays

4.Recursion

5.Depth First Search - As we traverse from one block to another and again back trace, we will be using DFS

Code:

BoggleBoard(Already given)

BoggleGame(Already given)

BoggleSolver

Attributes we used in BoggleSolver:

1. TrieSET dictionary

2. HashSET Validwords

**Methods Used :**

**BoggleSolver**(String [] dictionary) - this is the constructor where we are adding strings

into the dictionary.

**Iterable<String> getAllValidWords**(BoggleBoard board):- Boggle board is passed as an argument and all possible and meaningful words are found out from the board uding DFS.

* 1. We first initialise the valid word hashset here and then append the valid words into this hashset by passing it to the traversal method
  2. Here we will be initializing a boolean array and pass those arguments to dfs algorithm method called traversal.
  3. This finally returns the hash set which is an iterable that contains all the valid words formed from the boggle board and present in the dictionary also.

**Void traversal** (BoggleBoard,row,column,boolean array,hash set, Prefix):

* 1. This is a private method which is the dfs backtracking algorithm.
  2. This method checks some conditions and calls itself (recursion).

**Int ScoreOf**(String word):

* 1. It returns the score for a word which is a valid one and there in the dictionary and returns the score to the word only if its length is greater than 2.

Complexities:

DFS (Traversal) - O(Vertices + Edges)

hasPrefix() - O(N)

HashSet - O(1)

getAllValidWords() = O(N) - due to recursion

scoreOf () - O(N)