CMSC 142 - DESIGN AND ANALYSIS OF ALGORITHMS
2nd SEMESTER AY 2015-16
LABORATORY PROJECT SPECIFICATIONS
BOGGLE SOLVER

BACKGROUND - THE WORD GAME BOGGLE

Boggle is a word game designed by Allan Turoff [1] and originally distributed by Parker Brothers [2]. The game is played using a plastic grid of lettered dice, in which players attempt to find words in sequences of adjacent letters.

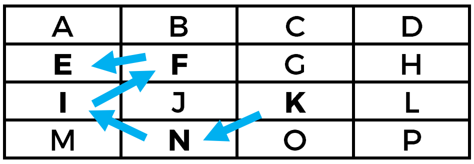
The game begins by shaking a covered tray of 16 cubic dice, each with a different letter printed on each of its sides. The dice settle into a 4×4 tray so that only the top letter of each cube is visible. After they have settled into the grid, a three-minute sand timer is started and all players simultaneously begin the main phase of play [3]. Each player searches for words that can be constructed from the letters of sequentially adjacent cubes, where "adjacent" cubes are those **horizontally**, **vertically**, and **diagonally neighboring**. Words must be **at least three letters long**, may include singular and plural (or other derived forms) separately, but **may not use the same letter cube more than once per word**. After three minutes have elapsed, all players must immediately stop writing and the game enters the scoring phase.

As an example, consider the 4x4 tray below with the 16 cubic dice randomly placed already:

A	B	C	D
E	F	G	H
I	J	K	L
M	N	O	P

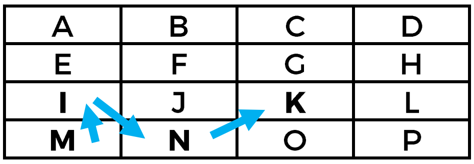
Depending on a dictionary used, there are approximately 20-35 possible words that can be found on the tray. Some examples of the words are shown below:

Word 1: KNIFE



A	B	C	D
E	**F**	G	H
I	J	**K**	L
M	**N**	O	P

Word 2: MINK



A	B	C	D
E	F	G	H
I	J	**K**	L
M	**N**	O	P

In the original word game boggle, the cubes involved with letter ‘Q’s are always followed by a ‘u’, thus have labels ‘Qu’. In our version of word game boggle, this is not followed. Every cube/cell only has a single character.

THE BOGGLE SOLVER

The Boggle Solver is a solver for the word game boggle which enumerates all the possible valid words in a given boggle 4x4 tray instance. It only considers the words with length of at least three. It refers to a dictionary of words when checking for valid words. There are several approaches for enumerating all the possible valid words and one of them is the backtracking approach.

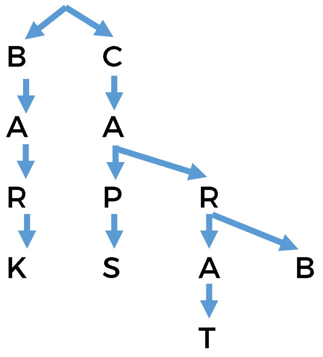
CHECKING FOR A VALID WORD USING TRIES

During its enumeration of possible words, the boggle solver should have a way of checking if a given string is a word in a given dictionary. The fastest way to check if a given string is a valid word or not is by using a Trie. Given a dictionary, all the words are first inserted in a trie. A given string is easily verified if it is in the trie or not. An example is given below.

Sample Dictionary:

BAR
BARK
CAR
CARAT
CARB
CAP
CAPS

Trie of the Dictionary:



Input String 1: CARAT (valid)
Input String 2: CARA (invalid)
Input String 2: BARBIE (invalid)

The input string 1 is a valid word since it is found in the trie. Input strings 2 and 3 are not valid words.

THE PROJECT

The CMSC 142 project requires groups made up of up to 3 (4, if the lab class size is not divisible by 3) students to create a program that will solve the word game boggle by providing all possible words and count them. The input will be read from a text file with the following format:

```
2          // number of boggle tray instances
4          // dimension of tray 1
A B C D    // letters on tray 1
E F G H
I J K L
M N O P
4          // dimension of tray 2
Z Y X W    // letters on tray 2
V U T S
R Q P O
N M L K
```

Students are **required to use the iterative backtracking algorithm** discussed in the laboratory as basis for creating their boggle solver. The code **may be modified as necessary in order to accommodate the nature of the Word Game Boggle**. The solver must be able to **find all possible valid words for each boggle instance in the input, as well as the correct number of possible valid words**. These results must be **printed to an output text file while there is no user interface** to view them. The format of the said output text file is up to you. Students are also **required to use the trie data structure to store the whole dictionary of words and to check whether a given string is a valid word or not**. The dictionary that will be used for this project will include the words form this text file of 178,691 words: goo.gl/VB6vSO [4].

There will be several milestone presentations so that your laboratory instructors will be aware of the progress you are making in your project, and to ensure that you will not leave the project unattended until the last possible moments of the semester. The details of these milestone presentations are:

- **Milestone 1 (Week of May 2-6, 2016)**
  - Must be coded using C.
  - Can look for all valid words with at least 3 letters when solving a 3x3 tray of 9 letters.
  - Can output the correct number of valid words.
  - Uses linear search when checking whether a given string is valid or not

- **Milestone 2 (Week of May 9-13, 2016)**
  - May be coded using any programming language at your disposal.
  - All requirements from milestone 1.
  - Can look for all valid words with at least 3 letters when solving a tray with more than 3x3 as dimension.
  - Can detect whether a tray has no valid words.

    - Uses the trie data structure to store the whole dictionary and to check if a string is a valid word or not.

- **Milestone 3/ Project Presentations (May 20, 2016)**
    - All requirements from milestones 1 & 2.
    - Working user interface that will show all possible solutions for all puzzles specified in the input.
    - Having a playable user interface will score more points than having a view-only user interface.
    - Having a better running time performance may score more points.
    - Additional features not mentioned in any milestone (may or may not have impact on your points).

GRADING

There will be a peer evaluation component to the project grade that will be used as a multiplier to compute the final project grade. A perfect peer evaluation score will grant you the full points awarded to your project, but anything less than a perfect peer evaluation score will be reflected in a lower final project grade. For example:

Raw Project Grade	**Peer Evaluation**	**Final Project Grade**
100%	100%	100%
100%	50%	50%

REFERENCES

[1] Murphy, Mary Beth (1978-12-09). Toy Designer's Creations Boggle the Mind.

[2] Hinebaugh, Jeffrey P. (2009), *A Board Game Education*

[3] BOGGLE Game Instructions - Hasbro,

[4] github.com/cviebrock/wordlists