CSE 2010, Term Project, Spring 2025
Due Tue Apr 22 at the start of your lab section; Canvas: Term
Project: Initial
Due Tue Apr 29 at the start of your lab section; Canvas: Term
Project

The game of Boggle asks a player to identify words from a 4x4
board of letters (printed on dice). A valid word:

- has at least 3 letters,
- has successive letters that are horizontally, vertically or diagonally adjacent on the board,
- can use each letter on the board at most once, and
- is in the dictionary.

Since "Q" is usually followed by "U", "Q" on the board means
"QU", which is counted as two letters. How would you design a
system that can find words correctly and quickly?

The challenge of the term project is to design and build BogglePlayer, a system that can identify words from a 4x4 board. In
addition to a time limit of 3 minutes, the number of returned words
is limited to at most 20 and the number of points for each unique
valid word is $(length(word) - 2)^2$ [corresponding negative points
for invalid/duplicate words with at least 3 letters; -1 point for each
word with fewer than 3 letters]. The total number of points is the
sum of the word points. We desire a system that achieves more
points, is fast, and has low memory usage, which can be measured
by:

1. total number of points for each 4x4 board
2. time used to identify words from each board
3. number of bytes used in the memory
4. $score = points^2/\sqrt{time * memory}$

All the measurements will be obtained on code01.fit.edu. Since
this course focuses on data structures and algorithms (not concurrency/parallelism), we would like you to focus on designing/selecting data structures and algorithms to improve performance. That is, using threads is NOT desirable for this term
project.

Program Files: Among your program files, you will proivde
BogglePlayer.java that supports at least:

- BogglePlayer(...) // initialization
- getWords(...) // return found words with board positions of letters

The initialization allows you to process the data and construct data
structures in preparation for playing Boggle. The time limit for the
initialization is 3 minutes. We will provide EvalBogglePlayer.java
(NOT TO BE MODIFIED) and Word.java (additional fields and
functions/methods can be added, but existing functions/methods
SHOULD NOT BE MODIFIED) to help you evaluate your system.

You may use any program segment/file from the textbook. You
may use any standard Java libraries (no additional installation).
However, you may *NOT* use program segments/files from other
sources. You may borrow ideas on data structures and algorithms
from other sources, but you need to clearly cite them at the top of
your program files and implement them on your own.

Input: Input is from the command-line arguments for EvalBogglePlayer.java:

- filename for a list of known words, one word per line
- optional seed for random number generator (sample seed: 13579)

Sample files are provided.

Output: Measured performance by EvalBogglePlayer.java goes
to the standard output (screen).

Presentation: Project presentations (about 10 minutes each)
will be during the lab on Apr 29. Create a presentation file following this outline:

1. Title, Group Name, and Members
2. Goal and Motivation
3. Initial Approach/Submission

   (a) Algorithms and supporting data structures
   (b) Additional input data and reasons
   (c) Ideas devised within the group
   (d) Ideas discussed in the course/book, cite them
   (e) Ideas borrowed from other sources, cite them

4. Final Approach/Submission

   (a) Changes in algorithms and supporting data structures
   (b) Changes in additional input data and reasons
   (c) Ideas devised within the group
   (d) Ideas discussed in the course/book, cite them
   (e) Ideas borrowed from other sources, cite them

5. Evaluation: Points, time, memory, and score: initial vs. final
   approach/submission.
6. Analysis

   (a) Why more/fewer points?
   (b) Why faster/slower (including time complexity—n words,
       each word has m characters)?
   (c) Why more/less memory?
   (d) Possible further improvements

Evaluation:

- Initial submission (20%) – including score and points
- Final submission (40%) – including score and points
- Presentation (20%, oral and written)
- Teammate evaluation (20%)
- Extra Credit: Final Submission has an average score above
  100, using less than 0.01 (10^{-2}) seconds and 10^8 bytes (~100
  MB), as measured by EvalBogglePlayer.java on code01.fit.edu.
  Average score of at least 101 earns one extra point, at least
  102 earns two extra points, ... 10 is the maximum number of
  extra points. Late submissions are not applicable.

Submission: Initial submision has BogglePlayer.java, additional
program files, and additional data files (if any).

Final submission has BogglePlayer.java, additional program
files, additional data files (if any), and presentation (preferably
in PDF).

To reduce issues on presentation day, we suggest you to send the
presentation file in PDF to the TA/GSA via email (in addition to
Submit Server).

Note the late penalty on the syllabus if you submit after the due
date and time as specified at the top of the assignment.