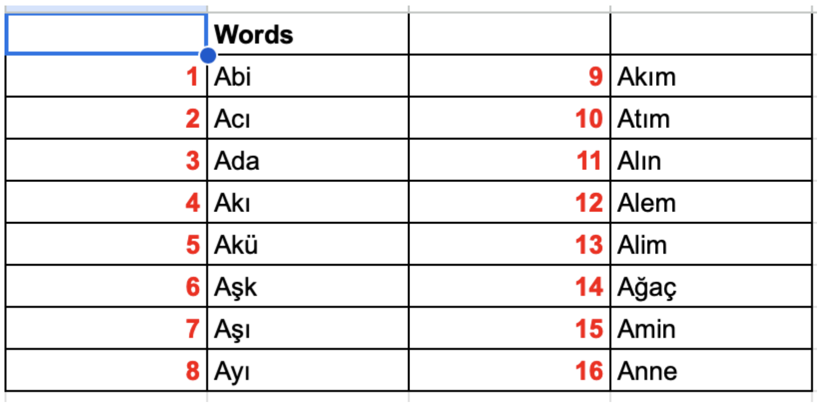
CS201 Project 3

Graph Puzzle Game

Please read the important notes part at the end carefully.

Project Description:

The aim of the project is to implement a graph puzzle game with provided graph structure on github.



	Words		
1	Abi	9	Akım
2	Acı	10	Atım
3	Ada	11	Alın
4	Akı	12	Alem
5	Akü	13	Alim
6	Aşk	14	Ağaç
7	Aşı	15	Amin
8	Ayı	16	Anne

Figure 1: Example Dictionary

Firstly, you need to download the provided dictionary on LMS. The dictionary has words in both Turkish and English so be careful. The dictionary consists of words of 3, 4 and 5 letters. These words are given in a mixed manner. You can check the example on top about understanding the concept of the dictionary. Dictionary will be on the '.txt' format so you need to read it from the file.

Puzzle Game Logic:

Objective:

Given a starting word and a target word, the objective is to find a sequence of words by changing only one letter at a time, such that each intermediate word in the sequence is a valid English word, and the final word in the sequence is the target word.

Rules:

1- Players can only change one letter in the word at a time.
2- Each intermediate word in the sequence must be a valid English or Turkish word.
3- The sequence of words should lead from the starting word to the target word.

Graph Structure:

Let's represent the relationships between words as a graph. Each node in the graph corresponds to a word, and there is an edge between two nodes if the corresponding words are connected by a valid letter change. Formally, the graph can be represented as

G = (V,E)

G = (*V*,*E*), where:

- *V* is the set of nodes representing words.
- *E* is the set of edges connecting words based on valid letter changes.

Players can traverse this graph by moving from one node to another along the edges, changing one letter at a time, until they reach the target word.

This formalization provides a clear framework for understanding the rules and structure of the word game, enabling a systematic approach to solving it.

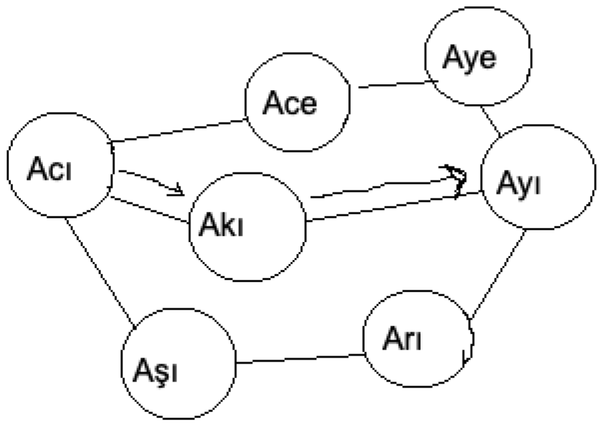


Figure 2: Example Graph Structure

As in the graph example above, a graph will be created for each of the words with 3, 4 or 5 letters given in the dictionary and this puzzle game will be played on it.

Weekly Tasks:

Week 1: Lexical Analysis and Graph Structure Establishment

During the initial week of the project, a comprehensive reading of the dictionary was conducted to identify word groups consisting of 3, 4, and 5-letter words. For each word group, a graph structure was established, forming the foundation for the subsequent phases of the project.

Week 2: Implementation of Breadth-First Search (BFS) Algorithm

In the second week, the focus of the project shifted towards the implementation of the Breadth-First Search (BFS) algorithm. This algorithm was chosen for its efficiency in exploring and traversing the established graph structures. The implementation was rigorously tested to ensure its correct functionality, conforming to the specifications outlined in the project requirements.

Week 3: Development of Shortest-Path Algorithm and Optimization

The third week marked a crucial phase in the project's development. Efforts were dedicated to enhancing the system by introducing a shortest-path algorithm. The primary objective was to identify and implement an algorithm capable of finding the shortest path between two words within a given word group graph.

Specifically:

1. Algorithm Development: A shortest-path algorithm was designed, considering the unique characteristics of the word group graph structures.

2. Implementation Refinement: The algorithm was integrated into the existing codebase, and optimizations were made to enhance its efficiency and performance.

3. Path Selection Logic: In situations where multiple paths existed between two words, logic was implemented to identify and select the shortest path.

4. Testing and Validation: Rigorous testing procedures were employed to verify the correct functioning of the shortest-path algorithm. This included testing with various word pairs and scenarios to ensure the algorithm's robustness.

By the end of the third week, the project aimed to achieve a fully functional system capable of efficiently finding the shortest path between words in the established word group graphs. This formalized approach ensured a systematic progression through the development stages, leading to a well-structured and optimized word game solution.

Project Structure:

- You will use the **C++ programming language** for this project.
- At first, please edit your github handle to be your name and surname so that we can identify you. If your github handle is “fullmetal-programmer”, it’s no good for us. If your name is “Yigit Demirsan”, please change it to something like “yigit-demirsan” or anything that can be identifiable. Also, please change your github profile name to your name and surname too. Then, use the following GitHub assignment, and create a repository for this project: https://classroom.github.com/a/MefcBfW0

Important Notes:

• You **MUST** come to the lab sessions to be graded for your work.

- You **MUST** accept the GitHub assignment with your name and surname only. Those who cannot be verified by us will receive 0.
- External library usage is forbidden.
- You can use data structures which are provided in the github.
- The code can only be written and be submitted to the github during the lab hours.
  However, you can practice and research outside of the lab hours.

If you have further questions, don't hesitate to contact me by my email at yigit.demirsan@ozu.edu.tr. Good luck and Başarılar!