

**UNIVERSITY INSTITUTE OF COMPUTING**

**PROJECT REPORT**

**ON**

**CROSSWORD PUZZLE**

**Program Name: BCA**

**Subject Name/Code: Data Structures (23CAT-201)**

**Submitted by: Submitted to:**

**Name: Shalini Raj Name: Dr. Shilpi Mittal**

**UID: 23BCA10319 Designation: Academic**

**Section: 23 BCA – 3(A) Co-ordinator**

**Group: 3rd**

**ABSTRACT**

**Introduction:**

**The Crossword Puzzle Generator is a C++ application designed to create and display crossword puzzles using a predefined list of words. The application focuses on randomly placing words into a grid while ensuring they fit according to crossword rules, such as overlapping letters. This project is an excellent exercise in programming logic, data structures, and algorithm implementation, providing both educational value and entertainment.**

**Technique:**

1. **Data Structures: Utilizes vectors to represent the crossword grid and to store the list of words.**
2. **Randomization: Implements random placement of words in the grid to create unique puzzles.**
3. **Validation: Checks whether a word can be placed in a specific location without conflicts.**
4. **Backtracking: Attempts to fill the grid efficiently by trying different placements for words.**

**System Configuration:**

* **OS: Windows 10 or Linux**
* **Processor: Intel Core i3 (minimum); Core i5 or higher recommended**
* **RAM: 4 GB (minimum); 8 GB recommended**
* **Development Environment: Any C++ IDE (e.g., Visual Studio, Code::Blocks) or**

**Visual Studio Code with a C++ compiler (GCC or Microsoft C++ Compiler)**

**SUMMARY**

**This project implements a crossword puzzle generator that randomly fills a grid with words from a specified list. The application checks for valid placements, displays the resulting crossword grid, and ensures that the words do not overlap incorrectly. This implementation serves as a foundational project for exploring more complex features in puzzle generation.**

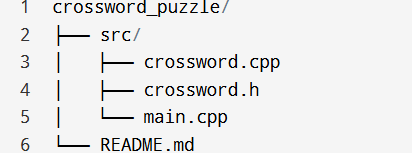
**INPUT:**

* **Word List: A list of words to be placed in the crossword (hardcoded in this example).**
* **Grid Size: The dimensions of the crossword grid (e.g., 10x10).**
* **User Interaction: Currently, the words are predefined, but the program could be extended to take user input for the word list.**

**Process:**

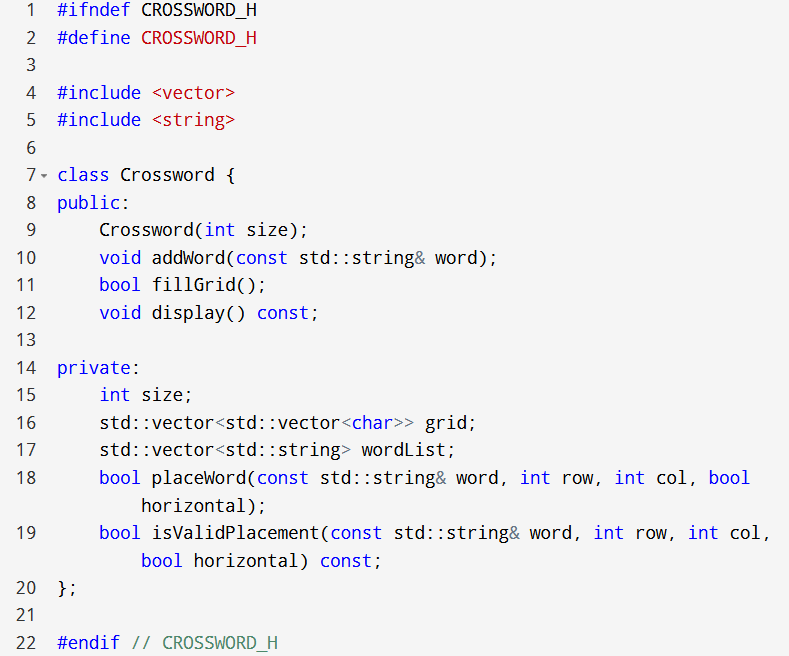
1. **Initialize the Grid: Create a 2D vector to represent the crossword grid, filled with placeholder characters (e.g. , '.').**
2. **Input Word List: Add words to the crossword from a predefined list.**
3. **Fill the Grid:**
4. **For each word in the list:**
   * + **Attempt to place the word randomly in the grid.**
5. **Check for valid placements:**
   * + **Ensure the word fits within the grid boundaries.**
     + **Ensure it does not overlap incorrectly with existing words.**
   * **If a valid position is found, place the word; otherwise, retry up to a certain number of attempts.**
6. **Display the Grid: Once all words have been placed, print the final crossword grid to the console.**
7. **End Program: The program concludes after displaying the grid.**

**Project Structure:**

****

**Src/main.cpp:**

****

**Src/crossword-h:**

**Conclusion:**

**This C++ project provides a foundational crossword puzzle generator. It illustrates basic concepts of programming and can be expanded with features like user-defined word lists, hints, or advanced placement algorithms. This project serves as a practical example for students and developers interested in game development and algorithm design.**

Output: