**Plagiarism detection system**

**Difficulty level**: Medium

Writers face difficulty in detecting plagiarism in their work, which leads to legal issues at times. This tool will help you analyze documents, implement algorithms to compare them with other text documents, and detect similarities within them.

**Learning outcomes**:

* Implementing string-matching algorithms for the identification of plagiarism
* Use of data structures like hash tables to store and manipulate data for comparison
* Optimizing algorithms to handle large volumes of text data efficiently
* Gain knowledge of data preprocessing, text normalization, and handling large datasets

**What it takes to execute this project**:

* Use a data structure like a hash table or trie to store and index document text.
* Implement algorithms for text preprocessing (tokenization, stemming, stop word removal).
* Utilize string matching algorithms (e.g., Knuth-Morris-Pratt, Boyer-Moore) for finding plagiarized text segments.
* Implement similarity measures like cosine similarity, Jaccard index, or n-gram matching.
* Use clustering algorithms (e.g., k-means, hierarchical) to group similar documents.
* Implement efficient data structures (e.g., suffix arrays, suffix trees) for pattern matching.
* Handle large document collections using techniques like shingling or locality-sensitive hashing.
* Provide a user interface for uploading documents and displaying plagiarism detection results.
* Integrate with external document sources or databases for comprehensive checking.

**Real-world application**:

* **Educational institutions**: They can use this application to look into academic papers and students’ assignments to detect plagiarism
* **Publishing house**: These houses need to publish articles and videos, and they can use the application to ensure the content is free from plagiarism
* **Content creators**: Content creators need to check for plagiarism in their content before submitting it to clients or releasing it on different platforms, and therefore, this application is useful for them
* **Legal house**: Lawyers use this application to analyze contracts, patents, or other legal documents for plagiarism or intellectual property violations

**Travel planner using Graph**

**Difficulty level**: Medium

This project aims to help travelers plan optimal travel routes between different locations. It uses graph data structures to represent locations as nodes and paths and the connecting roads or transportation routes as edges. This application will be able to calculate the shortest path between two locations, considering factors like distance, travel time, and cost.

**Learning outcomes**:

* Hands-on experience in working with graph data structures and algorithms
* Learn different types of graphs like directed, undirected, and weighted graphs
* How to represent and traverse graphs using techniques like adjacency lists or matrices
* Implement graph algorithms like Dijkstra’s algorithm, DFS (Depth-First-Search), and BFS (Breadth-First-Search) to find the shortest path between two locations
* Develop skills in data handling, input/output operations, and user interface design

**What it takes to execute this project:**

* Represent locations as nodes in a graph data structure, with edges representing travel routes.
* Assign weights to edges based on distance, travel time, or cost.
* Implement graph traversal algorithms like depth-first search (DFS) or breadth-first search (BFS) to find possible routes.
* Use shortest path algorithms like Dijkstra’s or A\* to find optimal routes based on distance/time/cost.
* Implement data structures like priority queues or heaps for efficient path-finding.
* Handle constraints like budget, travel mode, or stopover preferences.
* Integrate with external APIs for fetching location data, travel information, or real-time updates.
* Provide a user interface for entering source, destination, and preferences, and displaying planned routes.

**Real world application**:

* **Navigation systems**: Navigation systems like GPS devices and maps can implement this application to find the best roads for travelers and drivers
* **Logistics and supply chain industry**: They can use such systems to optimize their delivery routes for trucks to reduce cost and improve delivery time
* **Travel agencies**: These organizations can provide better services by planning efficient travel roads for clients considering different constraints with this application

**Cash flow minimizer**

**Difficulty level**: Medium

This cash flow minimizer project focuses on developing a program that will minimize the cash flow among a group of individuals who have borrowed money from each other. It will use graph algorithms with optimization techniques like the Min-Cost Max-Flow algorithm to settle debts in minimum transactions.

**Learning outcomes**:

* Learn multiple data structures and algorithms related to cash flow management and optimization problems
* Use of efficient data structures like trees or priority queues for storing and manipulating data-based transactions
* Gain experience in implementing sorting and searching algorithms and greedy optimization techniques
* Handling large volumes of data, input/output operations, and designing user interface

**What it takes to execute this project**:

* Represent cash flow data as a time series or sequence using an appropriate data structure (e.g., array, linked list).
* Implement algorithms to identify and categorize income and expense transactions.
* Use sorting algorithms (e.g., merge sort, quick sort) to order transactions chronologically.
* Implement data structures like stacks or queues to handle transaction dependencies and scheduling.
* Utilize dynamic programming or greedy algorithms to minimize cash flow by rescheduling transactions.
* Incorporate constraints like due dates, penalties, and interest rates.
* Provide a user interface for inputting transaction data and displaying optimized cash flow schedules.
* Handle error checking and input validation for transaction data.
* Consider integrating with financial databases or APIs for real-world data.

**Real-world application**:

* **Financial institutions**: These bodies can use such systems to optimize their interbank debt settlement process, reducing transaction costs and complexity
* **Corporate companies**: They can implement such a system to manage internal transfers and settlements between multiple departments or subsidiaries, enhancing cash flow management
* **Personal use**: You can use such a system for expense sharing with minimum cash flow

**Building a crossword puzzle game**

**Difficulty level**: Medium

Crossword puzzle games are often liked by children, and even parents allow them to play them to improve their vocabulary and language skills. This program will train students to build a crossword puzzle game that can generate and solve various puzzles. It will take a list of words, generate a crossword grid, and place the words in an interlocking pattern. It will also help users find and fill in the missing words depending on clues and the interlocking pattern of the grid.

**Learning outcomes**:

* Learn multiple data structures and algorithms related to string manipulation, search techniques, and pattern matching
* Use data structures like hash tables for storing and manipulating word lists
* Implementing backtracking algorithms or constraint satisfaction techniques to generate and solve different crossword puzzles
* Designing a user-friendly interface for users to interact with the system

**What it takes to execute this project:**

* Use a 2D array or matrix to represent the crossword puzzle grid.
* Implement a trie or prefix tree data structure to store and search for words efficiently.
* Utilize backtracking algorithms to find valid word placements in the grid.
* Implement constraint satisfaction techniques to ensure word intersections and grid validity.
* Use heuristics or scoring functions to guide the search process and optimize solutions.
* Integrate randomization algorithms for puzzle generation and variation.
* Implement data structures like stacks or queues for undo/redo functionality.
* Provide a user interface for displaying the puzzle, entering words, and interacting with game features.
* Handle input validation, error checking, and scoring mechanisms for gameplay.

**Real-world application**:

* **Play schools**: They can use this tool for building vocabulary and language skills among children
* **E-commerce platforms**: They can implement such games to attract customers to their platform and let them win cashback or discount coupons
* **Publishing houses**: Newspapers and magazines often print crossword puzzles for readers to improve their vocabulary, and with this tool, creating new puzzles regularly becomes easy