IDEA PRESENTATION

Submitted by
Aysha Naurin
Abhiram Ashok
Sreelakshmi K
Febin Nelson P

Guided by
Prof. Shibu Kumar

GENERAL TESTING FRAMEWORK FOR ACADEMIC PROGRAMS

IDEA

What is the Project About?

- An automated testing framework for evaluating student programming submissions.
- Provides test cases, error detection, and actionable feedback to streamline the assessment process.

PROBLEM STATEMENT

Challenges in Academic Program Evaluation:

- Manual evaluation is time-consuming and error-prone.
- Difficulty in identifying and explaining errors in student submissions.
- Lack of instant, personalized feedback for students to learn from their mistakes..

PROJECT OBJECTIVE

What the Project Aims to Achieve:

- Automate program evaluation by testing student submissions against predefined test cases.
- Detect and highlight errors with precision.
- Provide predictions for the cause of errors and suggest potential solutions.
- Reduce faculty workload and enhance the student learning experience.

FEATURES

1. Automated Testing:
    - Run student submissions against test cases and compare outputs.
2. Error Detection:
    - Pinpoint discrepancies between expected and actual outputs.
3. Feedback System:
    - Offer solutions and explanations for detected issues.
4. User Interface:
    - A user-friendly platform for instructors and students.

FEASIBILITY

Technical Feasibility

1. Compilation & Execution: Uses gcc compiler and Python subprocess for integration.
2. Error Detection: Employs gdb for error insights and segmentation fault detection.
3. Code Standards: Ensures coding standard adherence with cpplint and custom scripts.
4. Enhancement: Supports gradual addition of advanced features like error cause identification.

CONCLUSION

- An innovative solution to automate and improve academic programming evaluation.
- Reduces errors, increases efficiency, and offers valuable learning opportunities for students.
- Bridges the gap between manual effort and effective feedback in programming assessments.

C-BOOST

ENHANCING C WITH READY-TO-USE FUNCTIONALITIES

IDEA

What is C-Boost?

- A library offering ready-to-use sorting, searching algorithms, and data structures.
- Simplifies development by providing intuitive, pre-built functionalities.

PROBLEM STATEMENT

Challenges in C Development:

- Manually implementing essential algorithms (e.g., merge sort, binary search).
- Increased potential for bugs and inefficiencies.
- Time lost on repetitive tasks instead of focusing on high-level logic.

PROJECT OBJECTIVE

Goal of C-Boost:

- Abstract away algorithm complexity.
- Provide simple, ready-to-use functions (e.g., merge_sort(array)).
- Enhance developer productivity by reducing boilerplate code.

FEATURES

1. Sorting Algorithms:
    - Merge Sort, Quick Sort, etc.
2. Searching Algorithms:
    - Binary Search, Linear Search.
3. Data Structures:
    - Stacks, Queues, Linked Lists, Trees, Graphs.
4. Key Characteristics:
    - Efficient implementations.
    - Easy Integration with existing C projects.
    - Intuitive Function Calls for ease of use.

FEASIBILITY

Technical Feasibility

1. Programming Language: C (well-supported, portable, efficient).
2. Algorithms & Data Structures: Based on well-documented and proven techniques.
3. Implementation Plan: Modular design for easy extension and maintenance.

CONCLUSION

- C-Boost simplifies C programming by offering a comprehensive library of core algorithms and data structures.
- By using C-Boost, developers can save time, avoid errors, and streamline the development process.

AUTOEVAL

A VISUAL EVALUATOR FOR AUTOMATA AND TURING MACHINES.

IDEA

What is AutoEval?

- AutoEval is a visual simulator and evaluator for automata (FA, PDA) and Turing machines.
- Provides an interactive graphical interface for designing and evaluating state transitions and input strings.

PROBLEM STATEMENT

Challenges in Automata and Turing Machine Learning:

- Manually determining state transitions and outputs is time-consuming and error-prone.
- Difficulty in visualising automata behaviour.
- Evaluating and debugging Turing machine computations can be challenging without tools.

PROJECT OBJECTIVE

Aim of AutoEval:

- Visual interface for creating automata and Turing machines.
- Predict state transitions and outputs visually.
- Error-free, real-time evaluation for better learning.
- Simplify complex concepts through visualization.

FEATURES

1. Graphical Interface:
   - Design automata and Turing machines using a visual canvas.
2. Input String Evaluation:
   - Simulate and predict state transitions step-by-step.
3. Automata Support:
   - Finite Automata (FA), Pushdown Automata (PDA), and Turing Machines.
4. Key Characteristics:
   - User-friendly interface.
   - Accurate and real-time evaluation.

FEASIBILITY

Technical Feasibility

1. Platform & Framework: React for building the interactive, responsive user interface and Next.js for efficient server-side rendering.
2. Graphical Representation: Use React Canvas to render automata and Turing machine diagrams interactively.
3. Feature Expansion: Easy to extend with new automaton types and additional learning features.

CONCLUSION

- AutoEval makes automata and Turing machine analysis easier with an intuitive graphical interface.
- Eliminates the need for manual calculations and reduces errors during state evaluation.
- Serves as an effective tool for students and educators to better grasp computational theory.

THANK YOU