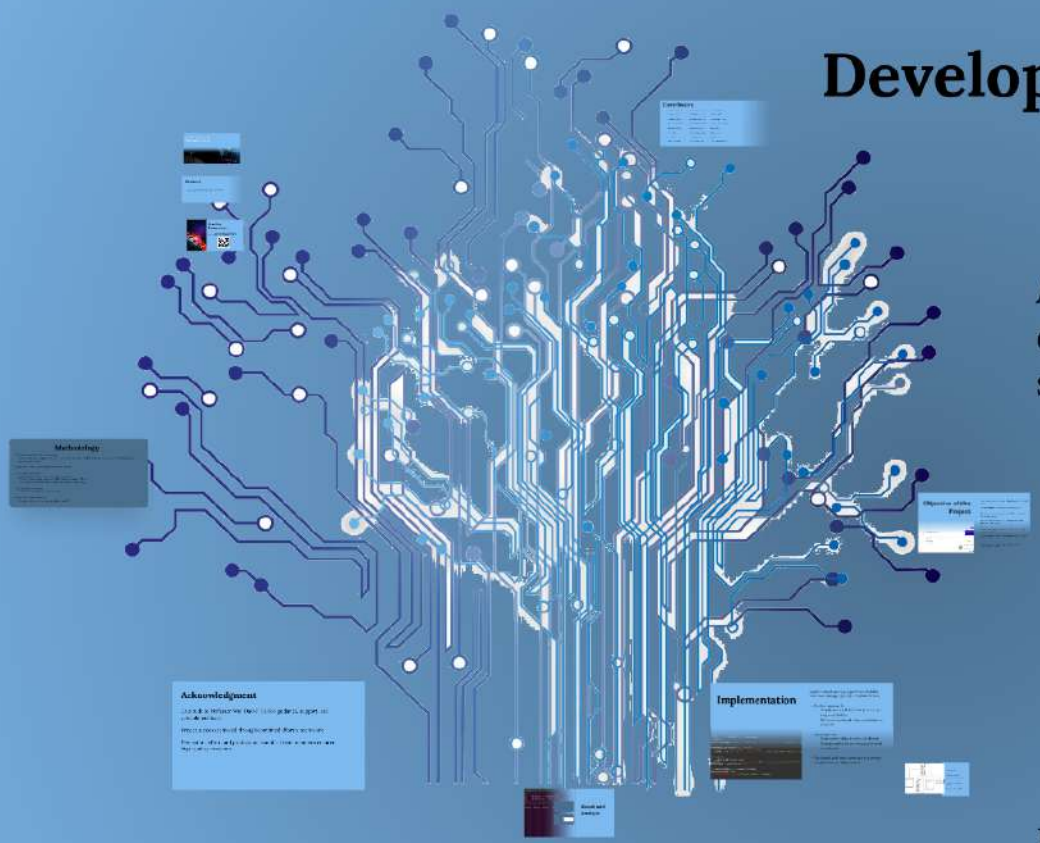


Course Project - 1
Developing a Memory Database with Sorting Algorithms

A comprehensive guide to creating a memory database with various sorting algorithms, utilizing real student data, and conducting performance analysis.

Purdue University northwest - Fall 2025

Contributors

• Matthews, Joshua	Matthe68@pnw.edu	Research Report
• Mishra, Amartya	Mishr259@pnw.edu	Python Code
• Mammai, Sreeja	Smammai@pnw.edu	Performance Tests
• Mhasawade, Siddhi	Smhasawa@pnw.edu	Python Code-Merge
• Mekala, Ruthvik	Rmekala@pnw.edu	Java & SQL
• Patel, Raaj	Pate2682@pnw.edu	Python Code
• Modi, Hirak	Modi54@pnw.edu	Slides
• Nalluri, Prasanth	Pnallur@pnw.edu	Java & Performance

Objective of the Project

- Implemented an in-memory database using linked lists.
- Loaded student-data.csv dataset for testing.
- Developed sorting algorithms: Bubble, Insertion, Merge, Quick sort.
- Extended system with SQL-like grammar and recursive CSV export.
- Added recursive export function to write sorted data back to CSV
- Conducted performance benchmarking in Ubuntu Linux
- Performance evaluated using Ubuntu Linux benchmarking tools.

Methodology

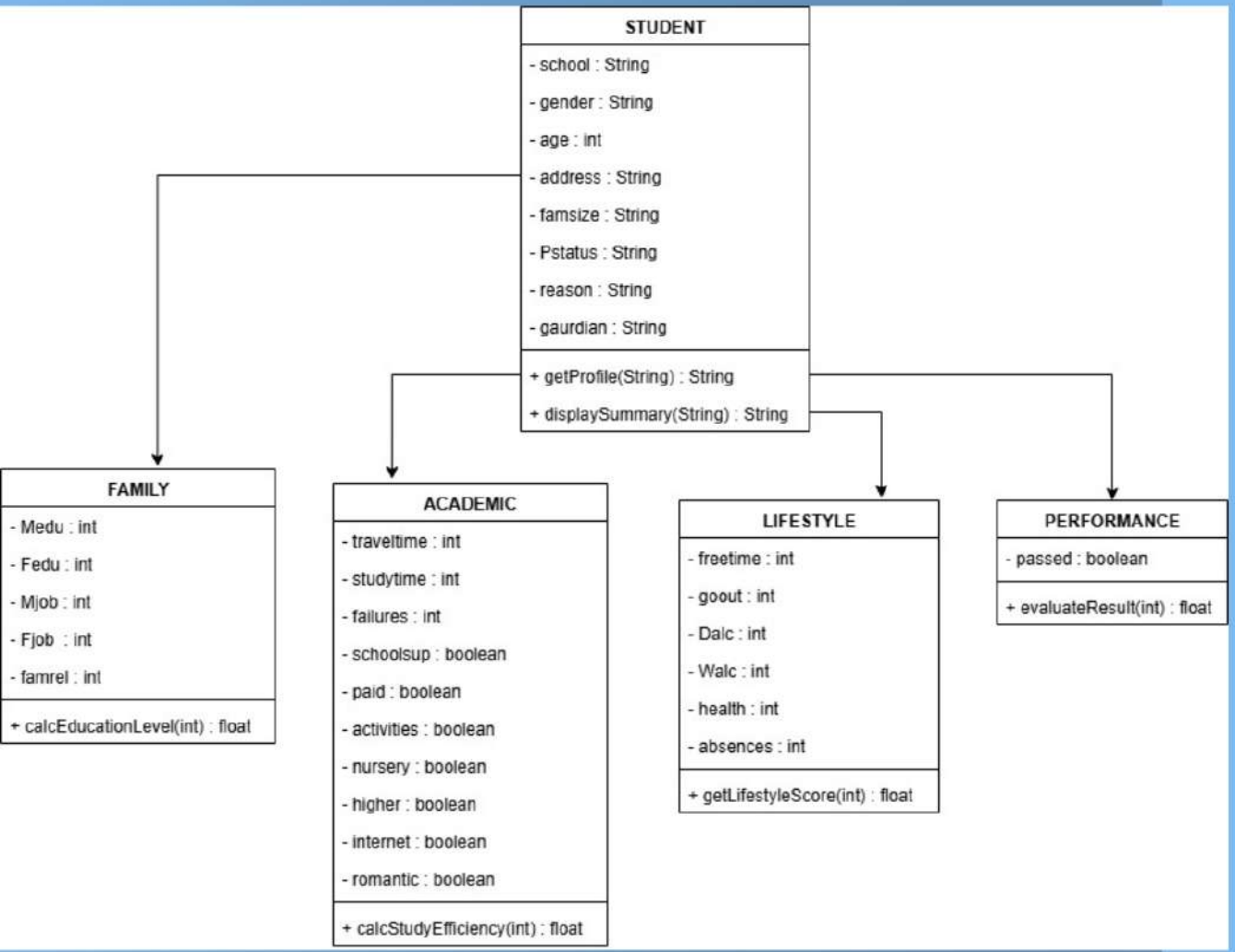
- Dataset Acquisition & Pre-processing
  student-data.csv (Kaggle) selected - structured, machine-readable, Parsed into tokens (ID, demographics, performance data)

- htop, time, iostat, pidstat for performance metrics

- Core Implementation
  Built singly linked list database in Python & Java
  Python - class-based, recursion, CSV export via built-in library
  Java - object-oriented, explicit node handling, strong typing

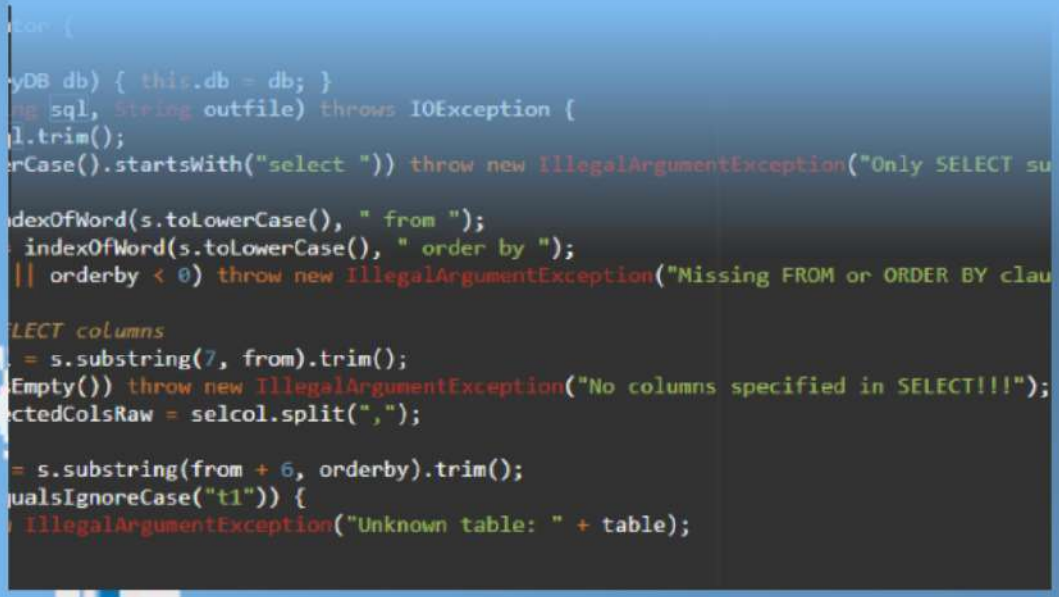
- SQL Grammar Extension
  Simulated SQL queries in toy database

- Recursive Export Function
  Recursive traversal - write sorted data to CSV



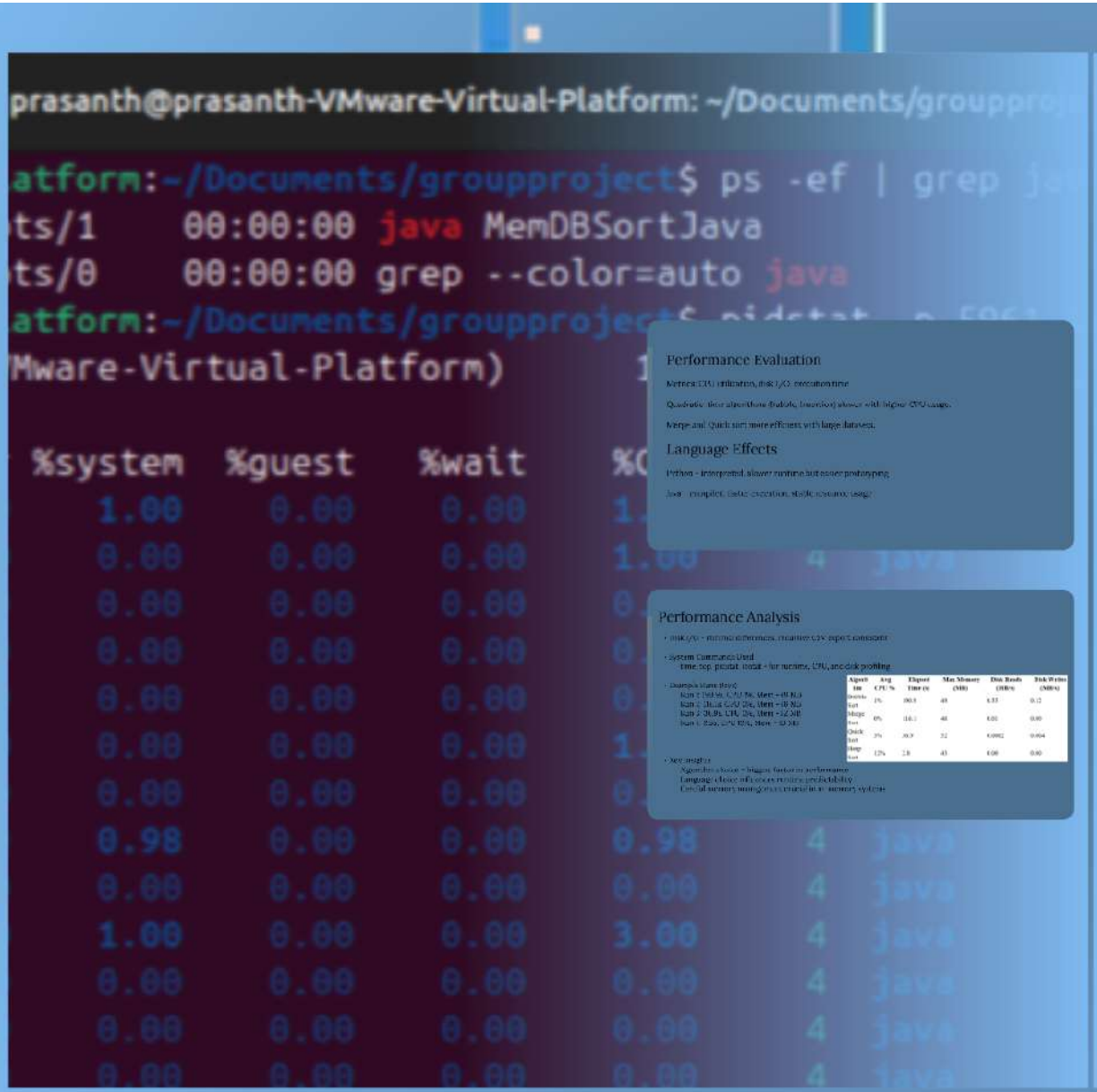
- Creating a UML Diagram Representation
- Visualized structure, process flow, and system integration
- Performance Benchmarking
- Compared Bubble/Insertion (quadratic) vs. Merge/Quick (efficient)
- Python vs. Java □ highlighted interpreted vs. compiled trade-offsMemory Database

Implementation



Implemented sorting algorithms (Bubble, Insertion, Merge, Quick) in Python & Java

- Python approach
  - Simple syntax & built-in structures - easy readability
  - Minimal overhead - fast validation of outputs
- Java approach
  - Structured, object-oriented design
  - Strong emphasis on encapsulation & modularity
- Validated with test cases (empty arrays, single element, duplicates)



Result and Analysis

Performance Evaluation

Metrics: CPU utilization, disk I/O, execution time

Quadratic-time algorithms (Bubble, Insertion) slower with higher CPU usage.

Merge and Quick sort more efficient with large datasets.

Language Effects

Python - interpreted, slower runtime but easier prototyping

Java - compiled, faster execution, stable resource usage

Performance Analysis

- Disk I/O - minimal differences, recursive CSV export consistent

- System Commands Used
  time, top, pidstat, iostat - for runtime, CPU, and disk profiling

- Example Runs (Java)
  Run 1: 190.9s, CPU 1%, Mem ~49 MB
  Run 2: 116.1s, CPU 0%, Mem ~46 MB
  Run 3: 36.9s, CPU 3%, Mem ~52 MB
  Run 4: 2.8s, CPU 12%, Mem ~43 MB

Algorithm	Avg CPU %	Elapsed Time (s)	Max Memory (MB)	Disk Reads (MB/s)	Disk Writes (MB/s)
Bubble Sort	1%	190.9	49	0.55	0.12
Merge Sort	0%	116.1	46	0.01	0.00
Quick Sort	3%	36.9	52	0.0002	0.004
Heap Sort	12%	2.8	43	0.00	0.00

- Key Insights
  Algorithm choice = biggest factor in performance
  Language choice influences runtime predictability
  Careful memory management crucial in in-memory systems

Acknowledgment

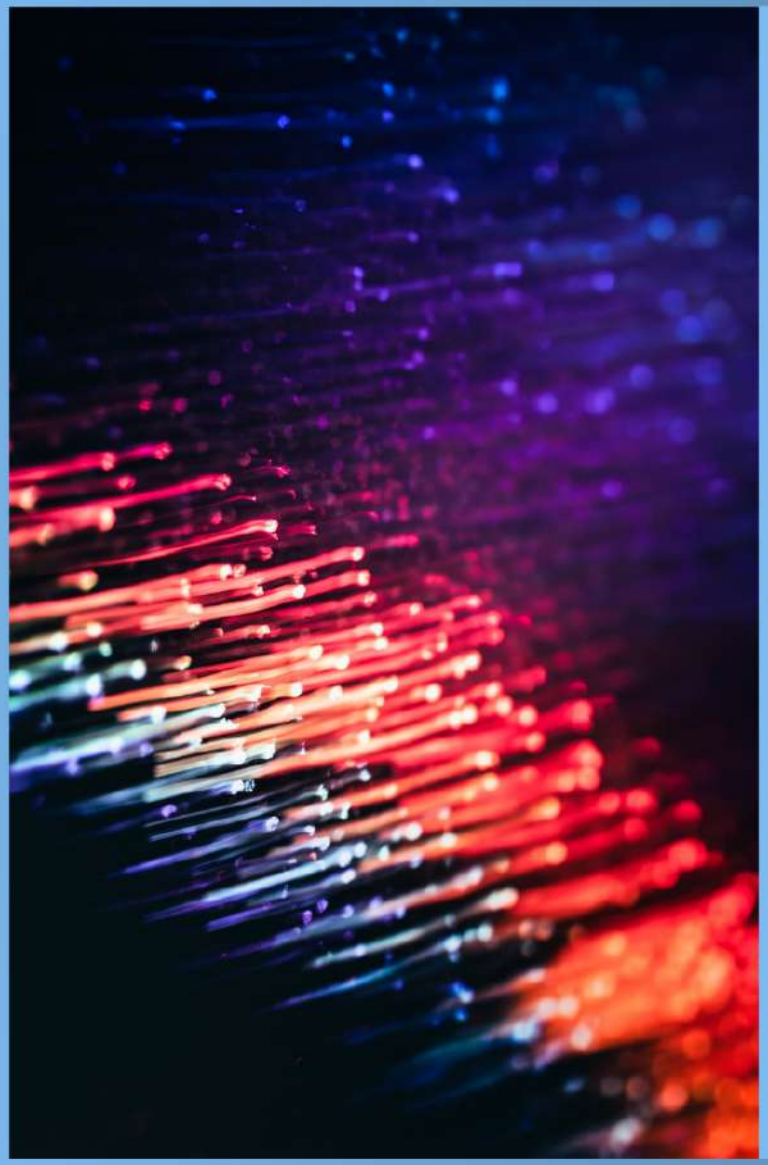
Gratitude to Professor Wei "David" Dai for guidance, support, and valuable feedback

Project success achieved through combined effort & teamwork

Dedication, effort, and professionalism of all team members ensured high-quality outcomes

Conclusion

- Successfully built a toy in-memory database system.
- Reinforced knowledge of linked lists, recursion, and algorithms.
- Benchmarked performance in Python and Java.
- Project emphasized collaboration and reproducibility.



Thank You!
Demonstration

Python: https://onlinegdb.com/bMV17hnXx
Java: https://onlinegdb.com/WA_w1BH9ZU