



Vivekanand Education Society's Institute Of Technology Department Of Information Technology

DSA mini Project

A.Y. 2024-25

STUDENT RECORD MANAGEMENT USING HASHING

Group Members

- NIKITA KHUSHALANI - 30
- KARTIK MAKHIJA - 31
- TANISHA NAWANI - 34

Mentor Name:

KAJAL JEWANI

Introduction to the Project

1

Problem Statement

Develop a student record management system using hashing techniques.

2

Objectives

Efficiently store and retrieve student data, provide search functionality.

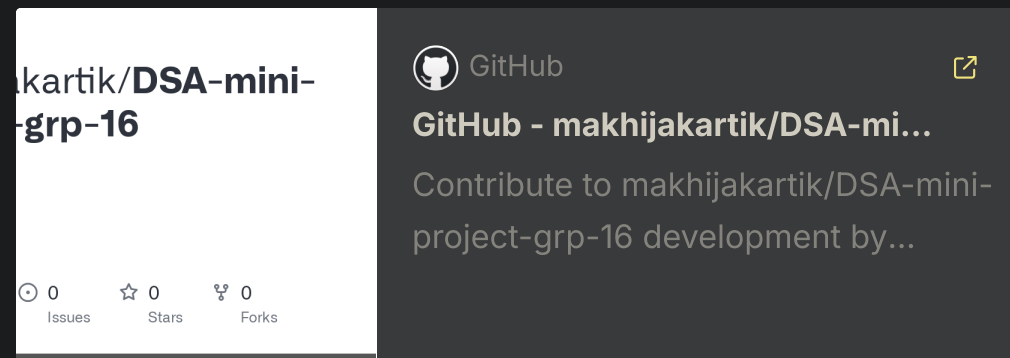
3

Scope

Manage student information, including ID, name, and other relevant details.

4

Code



makhijakartik/**DSA-mini-project-grp-16**

0 Issues 0 Stars 0 Forks

GitHub

GitHub - makhijakartik/DSA-mini-project-grp-16 development by...

Contribute to makhijakartik/DSA-mini-project-grp-16 development by...

Hashing

| | <u>Search</u> | <u>Insertion</u> |
|----------------|---------------|------------------|
| Unsorted array | $O(n)$ | $O(1)$ |
| Sorted array | $O(\log n)$ | $O(n)$ |

Linear Binary

BST $O(\log n)$ $O(\log n)$

Requirement: $O(1)$ for everything
Searching in $O(1)$ (Unsorted Array)

Solution 1: Direct Addressing

Let every element has a key and a value.

Ex: 100 employees \Rightarrow (key, value)
Each employee has
 \downarrow \downarrow
Let it be emp ID Let it be a name

Emp ID \rightarrow 5 Digit Integer
Eg 19047

Key acts as an index of array
 $arr[19047] \rightarrow$ Akshay

System Requirements

Hardware

Standard desktop or laptop computer, with sufficient memory and storage.

Software

C programming language, IDE (Integrated Development Environment).

Data Structures

Hash table, student record structure, linked list (for collision handling).

Concepts

Hashing, linear probing, string manipulation, input validation.



Time and Space Complexity



Time Complexity

$O(1)$ for average case, $O(n)$ for worst case (full hash table).



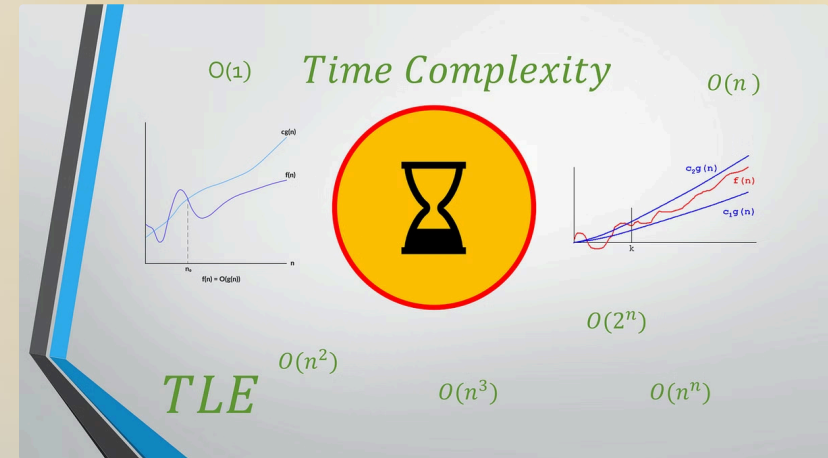
Space Complexity

$O(n)$ for the hash table, where n is the number of students.

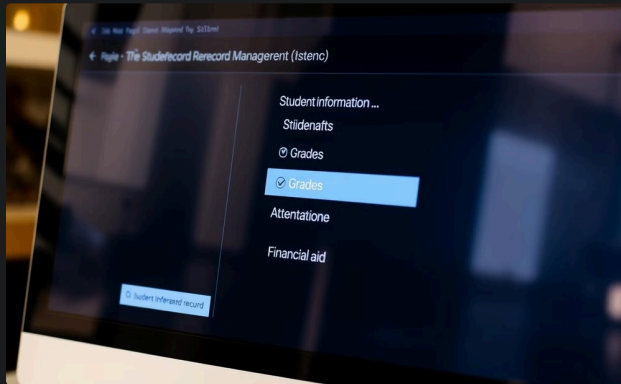


Efficiency

Hashing provides efficient storage and retrieval of student records.



Implementation



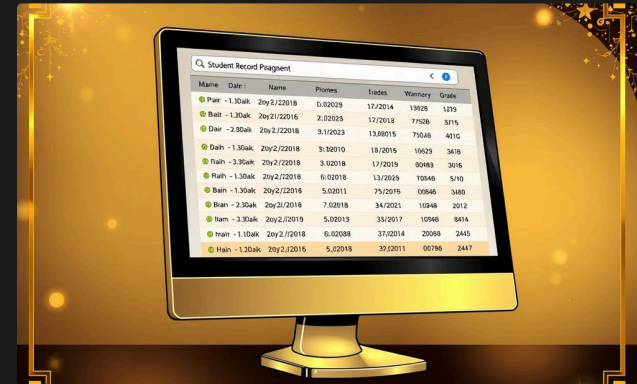
Main Menu

Allows users to add, display, and search student records.



Add Student

Prompts the user to enter student ID and name, then adds the record.



Search Student

Enables users to search for students by ID or name.

Algorithm Explanation

1 Hash Function

Compute the index in the hash table using the student ID.

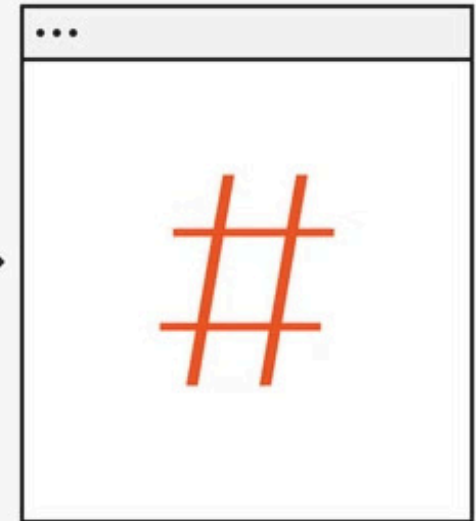
2 Collision Handling

Use linear probing to find the next available slot in the hash table.

3 Search Functionality

Implement search by ID and name, using the hash table and linear probing.

Hashing Algorithm



Hash Function

Gantt Chart

| | | |
|-----------------------|--------|--------|
| Project Planning | Sep 2 | Sep 8 |
| System Design | Sep 9 | Sep 15 |
| Implementation | Sep 16 | Sep 25 |
| Testing and Debugging | Sep 26 | Oct 2 |
| Documentation | Oct 3 | Oct 6 |



Challenges and Solutions

1

Collision Handling

Implemented linear probing to resolve collisions in the hash table.

2

Input Validation

Developed functions to validate student ID and name input.

3

Memory Management

Optimized memory usage by dynamically allocating student records.



CONCLUSION

The project successfully implemented a student record management system using hashing techniques. The system demonstrates efficient storage and retrieval of student data, providing essential functionality for managing student information



References

- **Journal Details**: International Journal of Computer Science and Education Technology, Volume 10, Pages 46-50
- **Citations Count**: 0
- **Keywords**: Hash Table, Student Records, Data Structure, Collision Handling

2. Summary

This paper describes a system designed to manage student records efficiently using hash tables. The system focuses on quick data storage and retrieval, utilizing a technique called linear probing to resolve collisions—situations where two student IDs hash to the same location. Key features include adding, displaying, and searching for student records, along with checks to ensure that the data entered is correct. The design is aimed at being straightforward, making it suitable for small to medium educational institutions.

3. Objectives

- Develop an efficient system for managing student records.
- Ensure that user inputs are validated to maintain data integrity.
- Provide functionalities for adding, viewing, and searching student records.



PDF file



Research paper.pdf

134.6 KB

<https://ieeexplore.ieee.org/document/9510144>

