**Students Accounts’ Management**

**Data Structures Project Description**

The project accounts’ management is an application to simplify the management of students’ financial records of a school. Add, remove and maintain financial records of students from class 1 to 10.

Provides financial reports to the school management as per the requirement.

**Scope**

* The project will help the accounts department in fast billing.
* This project will enable the accounts department to maintain a great database of all the students in the school.
* Project will enable to see report regarding students with fee overdue.

**Implementation**

* Information regarding students is stored using structures in a balanced binary tree.
* The data fields of the structure are name of the student (a character array), roll number of the student (integer), flag (an integer) which keeps track of the fees paid by the student.
* Name and roll number of the students are stored using text files. There are totally 10 text files. (one for each class)
* We read the data from text files and use it in our program.

**Functions used:**

* **update\_dataS () : Updates the data by promoting students each year while adding new admissions and removing outgoing students.**

Firstly, asks whether a new student is getting admitted or a student is leaving the school.

If choice is 1:

It asks for the name of the student and the class the student is joining. It calls insert() function and inserts the student into the tree.

If choice is 2:

It asks for the name of the student, roll number and the class the student is leaving. It calls the delete() function and the student’s record.

Once this is done, it promotes the student to the next academic year.

* **mid\_year() : Updates the data in case of midyear admissions and removing outgoing students.**

Firstly, asks whether a new student is getting admitted or a student is leaving the school.

If choice is 1:

It asks for the name of the student and the class the student is joining. It calls insert() function and inserts the student into the tree.

If choice is 2:

It asks for the name of the student, roll number and the class the student is leaving. It calls the delete() function and the student’s record.

* **fee() : Checking for valid student fee payment, display and update student records.**

Asks for class and name of the student, updates the flag variable to 1 and prints fees paid.

* **display(): Displays all students’ data.**

Asks for a class.

Displays name and roll number of all the students of that class.

* **search() : Searches for data of a student based on name.**

Asks for the student’s class and name and searches for the student by traversing the tree.

If found, prints student found.

Else, prints not found.

* **overdue() : Displays list of all students with fee overdue.**

Asks for a class.

Checks if flag is 0 (fee not paid), then displays list of all students of that class with fee overdue.

* **insert():** This function adds a new student to a class in alphabetical order.
* **delete():** This function removes a student from a class.
* **create():** Reads the text file and creates a node for each data.
* **extract():** Calls create() function and creates a binary search tree for each class.

**Limitations**

* Ignoring failure of students
* It’s manually driven. Not auto updated.