User Guide: Student Grade Management System

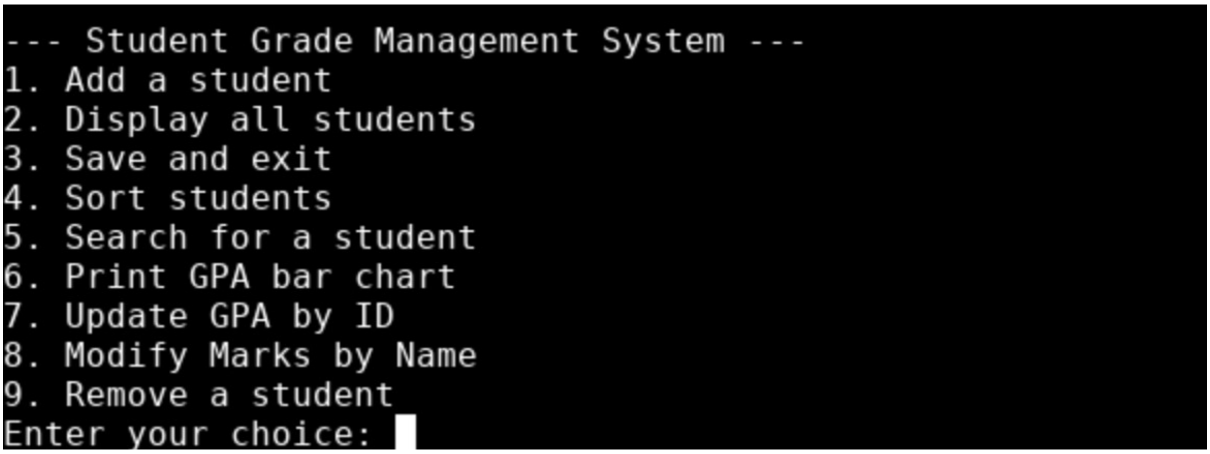
Compilation:

1. Ensure all files (main.c, student.c, student.h) are in the same directory.
2. Compile using: gcc main.c student.c -o output
3. Run the program:

   ./output

Program Menu & Usage :

Upon running the program, you will see the following menu:



```
--- Student Grade Management System ---
1. Add a student
2. Display all students
3. Save and exit
4. Sort students
5. Search for a student
6. Print GPA bar chart
7. Update GPA by ID
8. Modify Marks by Name
9. Remove a student
Enter your choice: 
```

1. Add a Student

- Prompts for:
  - 7-digit Student ID (unique)
  - Name
  - Grades for 5 subjects (0-100)
- Automatically calculates and stores GPA.

2. Display All Students

- Shows a formatted table with:
  - Student ID, Name, GPA, and individual subject grades.

3. Save and Exit

- Saves all student records to students.csv and exits the program.

4. Sort Students

- Options:

1. By ID
2. By Name
3. By GPA

- Displays the sorted list immediately.

5. Search for a Student

- Enter a Student ID to view details (ID, Name, GPA).

6. Print GPA Bar Chart

- Options:

1. All Students (shows GPA as a bar chart)
2. Single Student (shows subject grades as a bar chart)

7. Update GPA - by ID

- Enter a Student ID, then update subject grades.
- Automatically recalculates GPA.

8. Update GPA - by Name

- Enter a Student Name, then update subject grades.
- Automatically recalculates GPA.

9. Remove a Student

- Enter a Student ID to delete their record.

Expected Input and Outputs

1. Student Identification

- Each student in the system is assigned a **unique numeric ID.**
- This numeric ID helps in data operations such as updating, deleting, or searching for specific student records.

2. Academic Records Management

- The system allows storing and managing **grades for courses** that students have already registered in.

3. Menu-Driven User Interface

- The system is navigated through a **menu with options numbered from 1 to 8**, offering easy access to core features.

Known Strength

- No duplicate IDs allowed.
- Grades must be between 0 and 100.
- CSV file must follow the exact format (corrupt files may cause errors).

Troubleshooting

- Invalid ID: Ensure the ID is a 7-digit number.
- Student not found: Verify the ID or name exists.
- Error saving/loading CSV: Check file permissions or formatting.
- Memory leaks: If using Valgrind, ensure all functions free memory properly.

Example Workflow

1. Add Students:
    - Enter IDs, names, and grades for multiple students.
2. Display Students:
    - Verify all entries are correct.
3. Sort by GPA:
    - Check if sorting works (highest GPA first).
4. Save & Exit:
    - Confirm students.csv is created/updated.
5. Reload Data:
    - Restart the program to ensure CSV loading works.

Conclusion

This system provides an efficient way to manage student grades with dynamic memory usage, file persistence, and visualization. Follow the instructions carefully to avoid input errors.