Project Title : Quiz Result Analyzer

☐ Table of Content:

1.Introduction:

In the digital world search engines and data organization play a vital role. This mini project **“quiz result analyser” ,**simulates a basic search engine by sorting multiple student records,sorting them by using Marks and Time ,and also allowing users to search the result by using student name efficiently.

2.Team Members:

SNO	NAME	ENROLL NO	RESPONSIBILITY
1	Vikas Marepally	2403031460939	Searching algorithm by Name
2	Harish Seepana	2403031460811	Displaying functions
3	Ritesh Reddy	2403031460304	Main function and declarations
4	Sai Chandra	2403031461225	Testing and validation
5	Kalyani Annem	2403031460288	Sorting by Marks and its criteria
6	Abignya Reddy	2403031460533	Sorting by Time and its criteria

3.Objective:

To develop a C program that:

- Stores list of student records.
- sorting student rank based on student marks by using sorting algorithm.
- sorting student rank based on student time taken by using sorting algorithm.
- searching for a specific student rank by using linear search.

4. ?Problem Statement:

Developing a **Quiz Result Analyser System** that processes the performance of students in an online quiz each student has a **name, score(marks)** and **time taken** to complete the quiz. This project aim is to **sort student rank based on Marks and time taken** by the student and also to **search specific student rank by using student name** in searching algorithm.

5.□System Requirements:

Software:

- Turbo C++/dev C++
- Any Text Editor (VS code)

Hardware:

- Minimum 2GB RAM
- 1.0 GHZ Processor
- Laptop / keyboard&monitor

6.□Data Structures Used:

- Arrays

  Used array of structures to store data of multiple students.

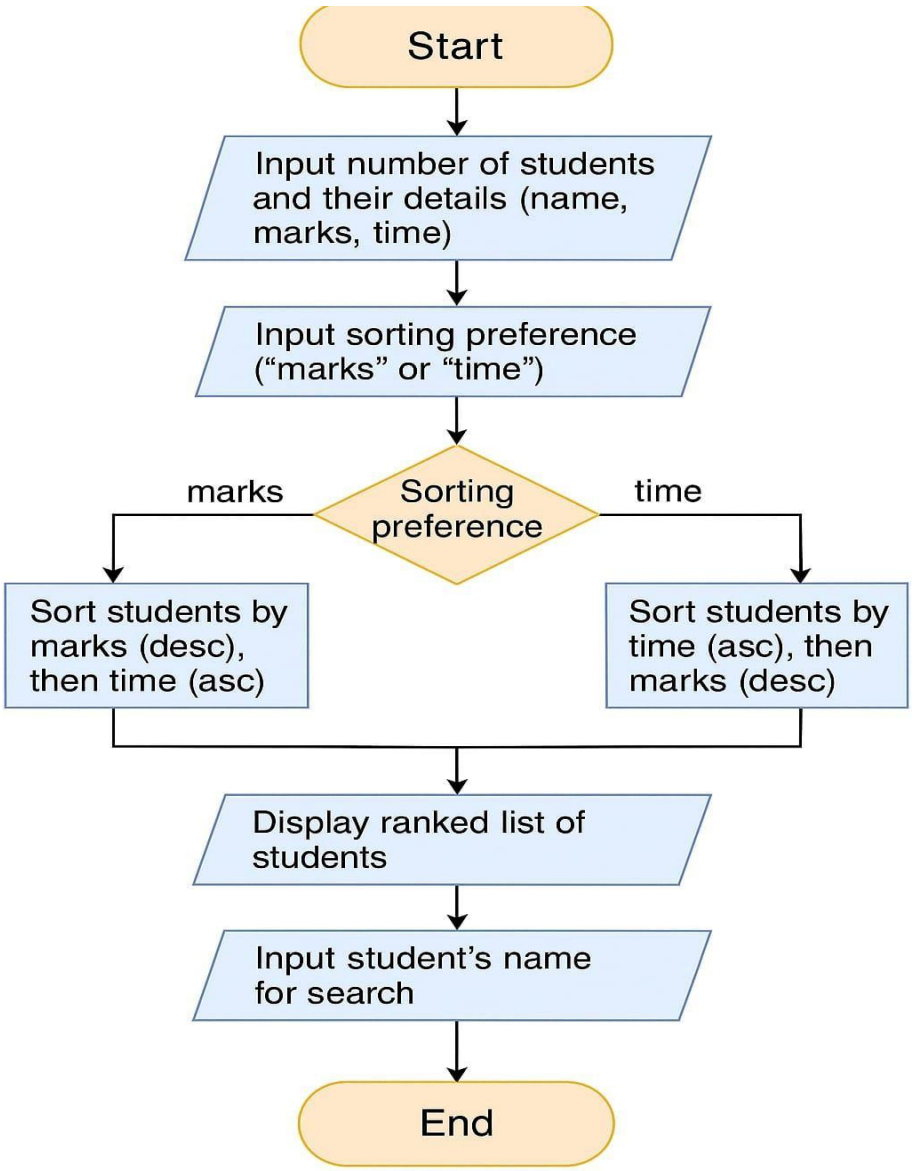
- Sorting(bubble sort)

  Sorting mechanism is used to sort student rank by using both marks scored and time taken by the student to complete the exam.

- Searching (linear search)

  searching mechanism is used to search a specific student rank by using student name.

Flowchart of the code:



7.Modules and Description:

MODULE	DESCRIPTION
Input Module	Accepts multiple records of Students(name,marks&time taken)
Display Module	Display all stored records
Sorting by Marks	Sorting students rank by marks (descending by marks then ascending by time)
Sorting by Time	Sorting students rank by time taken (ascending by time then descending by marks)
Searching Module	Searching specific student record by using student name (which is case-insensitive)

8.⚙Algorithms Used:

A. Sorting:- Bubble Sort(by marks):

// Sort students by marks (desc), then time

(asc) void sortByMarks(Student s[], int n) {

Student temp;

```
    for (int i = 0; i < n - 1; i++) {         for
(int j = 0; j < n - i - 1; j++) {             if
((s[j].marks < s[j + 1].marks) ||
                    (s[j].marks == s[j + 1].marks && s[j].timeTaken > s[j +
1].timeTaken)) {
                temp = s[j];
s[j] = s[j + 1];                s[j + 1]
= temp; } } } }
```

B.Sorting:-Bubble Sort(by time taken):

```
// Sort students by time (asc), then marks (desc)
void sortByTime(Student s[], int n) {     Student
temp;
   for (int i = 0; i < n - 1; i++) {         for (int j = 0;
j < n - i - 1; j++) {             if ((s[j].timeTaken > s[j
+ 1].timeTaken) ||                (s[j].timeTaken ==
s[j + 1].timeTaken && s[j].marks < s[j +
1].marks)) {
            temp = s[j];
s[j] = s[j + 1];
s[j + 1] = temp;
```

```
        }
    }
  }
}
```

C.Searching:-Linear Search(search specific student record by student name(case_insensitive):

```
// Case-insensitive search for a student by name int
searchByName(Student s[], int n, char target[]) {
  for (int i = 0; i < n; i++) {       if
(strcasecmp(s[i].name, target) == 0) {
          return i;
      }
  }
  return -1;
}
```

9.Sample input/output:

1.Input:

```
Enter number of students:6
Enter student 1 name:vikas
Enter student 1 marks:56
```

Enter time taken by student 1:57

Enter student 2 name:ritesh

Enter student 2 marks:83

Enter time taken by student 2:35

Enter student 3 name:harish

Enter student 3 marks:76

Enter time taken by student 3:51

Enter student 4 name:sai

Enter student 4 marks:68

Enter time taken by student 4:35

Enter student 5 name:kalyani

Enter student 5 marks:97

Enter time taken by student 5:48

Enter student 6 name:abignya

Enter student 6 marks:97

Enter time taken by student 6:50

Select sorting criteria(“marks or time”)? :Time/Marks

Enter student name:vikas

2.Stored Output:

6

Vikas , 56 , 57

Ritesh , 83 , 35

Harish , 76 , 51

Sai , 68 , 35

Kalyani , 97 , 48

Abignya , 97 , 50

Time/Marks

Vikas

3.Asking Sorting preference(by marks or time):

```
Sort by 'marks' or 'time'?
```

4.Sorting by marks(descending by marks and then ascending by time):

```
Students sorted by marks (desc), then time (asc):
Rank 1: kalyani - 97 marks, 48.00 min
Rank 2: abignya - 97 marks, 50.00 min
Rank 3: ritesh - 83 marks, 35.00 min
Rank 4: harish - 76 marks, 51.00 min
Rank 5: sai - 68 marks, 35.00 min
Rank 6: vikas - 56 marks, 57.00 min
```

5.Sorting by Time(ascending by Time and then descending by Marks):

```
Students sorted by time (asc), then marks (desc):
Rank 1: ritesh - 83 marks, 35.00 min
Rank 2: sai - 68 marks, 35.00 min
Rank 3: kalyani - 97 marks, 48.00 min
Rank 4: abignya - 97 marks, 50.00 min
Rank 5: harish - 76 marks, 51.00 min
Rank 6: vikas - 56 marks, 57.00 min
```

6.Searching by Student Name(Case-insensitive):

```
Enter name to search:
Student vikas found at rank 6
Marks: 56, Time taken: 57.00 min
```

10.☑Test Cases:

Test Cases	Input	Sorting or Searching	Expected Output

Case:1	6 Vikas , 56 , 57 Ritesh , 83 , 35 Harish , 76 , 51 Sai , 68 , 35 Kalyani , 97 , 48 Abignya , 97 , 50 Marks	Sorting by Marks	Students sorted by Marks (desc), then Time (asc): Rank 1: kalyani - 97 marks, 48.00 min Rank 2: abignya - 97 marks, 50.00 min Rank 3: ritesh - 83 marks, 35.00 min Rank 4: harish - 76 marks, 51.00 min Rank 5: sai - 68 marks, 35.00 min Rank 6: vikas - 56 marks, 57.00 min
Case:2	6 Vikas , 56 , 57 Ritesh , 83 , 35 Harish , 76 , 51 Sai , 68 , 35 Kalyani , 97 , 48 Abignya , 97 , 50 time	Sorting by Time	Students sorted by Time (asc), then Marks (desc): Rank 1: ritesh - 83 marks, 35.00 min Rank 2: sai - 68 marks, 35.00 min Rank 3: kalyani - 97 marks, 48.00 min Rank 4: abignya - 97 marks, 50.00 min Rank 5: harish - 76 marks, 51.00 min Rank 6: vikas - 56 marks, 57.00 min
Case:3	Vikas	Search by Name	Student vikas found at rank 6 Marks: 56, Time taken: 57.00 min
Case:4	Manoj	Search by Name	Student Manoj not found.

11.Future Enhancements:

⇛Store and Load Data from files

⇛Support for Floating Point Marks(marks in points)

⇛Improved Search(search partial name to search full name)

⇛Advanced Sorting Options(sort by alphabetical order)

⇛Graphical Representation

⇛Rank Assignments(assigning rank properly for ties)

⇛Multiple Quiz records(storing multiple records of one student)

⇛User Interface / Menu Systems (add newstudent,sort,search,display all,exit)

⇛ Export to Excel Format(export result to a .csv file that can be opened in excel or google sheets)

12.Conclusion:

This program acts as a basic functional **“QUIZ RESULT ANALYSER”** that sorts student data based on performance and enables efficient search by Name.It helps in ranking students based on both accuracy (Marks) and efficiency (Time taken), which can be useful in competitive assessments. This project successfully performs Basic Quiz Result Analysis.

13.References:

1. Data Structures classes by Jashwanth Sir
2. Chat-GPT
3. ByteXL-App

Thankyou