PROJECT REPORT

C PROGRAMMING

School of Computer Science

University of Petroleum and Energy Studies

PROJECT TITLE → Result Management System (with Ranking & GPA)

- Enter marks of multiple students.
- Calculate GPA/percentage.
- Rank students based on marks.
- Search student records.
- Generate report card to file.

SUBMITTED BY →

NAME→ Shaurya Singhal

SAP ID →590023883

BATCH →60

Instructor: Dr. Prashant Trivedi

Course Code: CSEG1041

INDEX

ABSTRACT

- **WHAT THIS PROJECT DOES**

  This project develops a result management system in the C programming Language that records student marks and automatically calculates total marks, percentage, GPA, and rank

- **WHY IT IS USEFUL**

  It simplifies the result calculation process by elimination manual errors saving time and providing a clear and accurate evaluation of student performance in a standard manner

- **HOW IT WORKS AT A HIGH LEVEL**

  The system takes student marks as input processes them using functions, arrays, and sorting techniques computes required values and displays the final results along with GPA and ranking in an easy-to-read format

PROBLEM DEFINATION

- **MANUAL RESULT CALCULATION IS TIME CONSUMING**
  in schools and colleges calculating total marks GPA and ranks manually takes a lot of time and effort

- **HIGH CHANCE OF HUMAN ERROR**
  Manual calculation often led to mistakes such as wrong totals incorrect GPA or incorrect ranking

- **DIFFICULT IN MANAGING MULTIPLE STUDENTS**
  Handling multiple student records at the same time becomes complex without a computerized system

- **CHALLENGES IN PRESENTING RESULT CLEARLY**
  Manually prepared result may not be well formatted or easy to understand

- **NO AUTOMATIC RANKING SYSTEM**
  Ranking student based on marks required sorting and comparison which is tedious to do manually

- **NEED FOR A STANDARDIZED GPA CALCULATION**
  Different teachers use different method a program ensures consistent and uniform GPA calculation

ALGORITHM

- Start
- Input number of students(num)
- Create an array of student structure of size num
- For each student (loop i = 0 to num – 1)
    1. Input student name
    2. Input student roll number
    3. Input marks of 5 subjects
- Calculate totalmarks & percentage for each student (loop i = 0 to num – 1)
    1. totalmarks = sum of all 5 subjects
    2. percentage = totalmarks/5.0
- Sort all student by total marks (descending order)
  Use bubble sort:
    1. compare marks of two students
    2. if smaller, swap them
    3. continue until fully sorted
- Assign ranks to sorted student
    1. first student gets rank = 1
    2. if total marks equal to previous → same rank
    3. else → rank = index +1
- Ask user if they want to search a student record(y/n)
  IF YES:
    1. input no. of student record user want to see
       FOR EACH STUDENT
        1. input roll number
        2. use searchstudent()to find matching roll

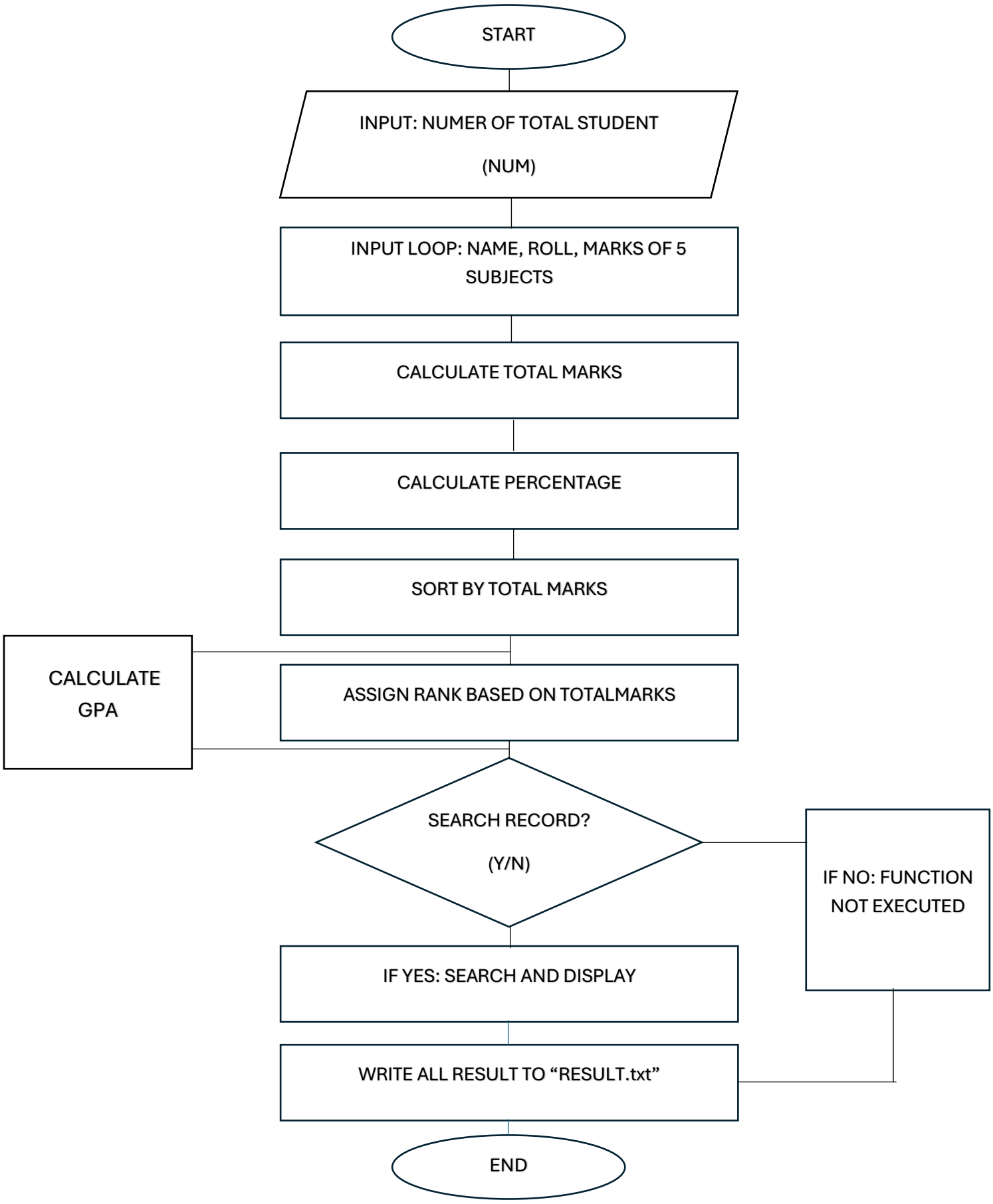
3.if found →print name, roll, percentage
4.else→print “student not found”

- Open file RESULT.txt for writing
  For each student (already sorted by rank):
    1. Write rank
    2. Write name
    3.Write roll number
    4.Write all subject marks
    5.Write total marks
    6.Write percentage
    7.Print a separator line

- Close the file

FLOWCHART

- **OVERALL FLOWCHART**



IMPLEMENTATION DETAILS

- **Data structure**

```c
struct subject{
    int C_PROGRAMMING;
    int LINUX;
    int EVS;
    int MATHS;
    int PROB_SOLVING;
};


struct student{
    char name[100];
    int roll;
    struct subject sub;
    int totalmarks;
    float percentage;
    int rank;
};
```

- **Read multiple students**

```c
struct student stu[num];


    for(int i = 0 ; i < num ; i++){
        char name[50];
        printf("enter the name of %d student: ",i+1);
        scanf("%99s",stu[i].name);

        printf("enter the roll number of the student: ");
        scanf("%d",&stu[i].roll);


        printf("enter marks of ( C_PROGRAMMING , LINUX , EVS , MATHS , PROB_SOLVING ): ");
        scanf("%d %d %d %d %d", &stu[i].sub.C_PROGRAMMING, &stu[i].sub.LINUX, &stu[i].sub.EVS, &stu[i].sub.MATHS, &stu[i].sub.PROB_SOLVING);


        printf("\n");
    }
```

- **Sorting and ranking**

```c
void SortByMarks(struct student stu[], int size){
    struct student temp;
    for(int i = 0 ; i < size-1 ; i++){
        for(int j = 0 ; j < size-1-i ; j++){
            if(stu[j].totalmarks < stu[j+1].totalmarks){
                temp = stu[j];
                stu[j] = stu[j+1];
                stu[j+1] = temp;
            }
        }
    }
}


void assignRanks(struct student stu[], int size){
    stu[0].rank = 1;
    for(int i = 1 ; i < size ; i++){
        if(stu[i].totalmarks == stu[i-1].totalmarks){
            stu[i].rank = stu[i-1].rank;
        }
        else{
            stu[i].rank = i + 1;
        }
    }
}

```

- **Search student record by roll**

```c
int searchstudent( struct student stu[] , int size , int roll){
    for(int i = 0 ; i < size ; i++){
        if(stu[i].roll == roll){
            return i;
        }
    }
    return -1;
}
```

- **Generate report file**

```c
FILE *fptr = fopen("RESULT.txt" , "w");

    for(int i = 0; i < num; i++){
    fprintf(fptr, "rank: %d\n \n",stu[i].rank);
    fprintf(fptr, "name: %s\n",stu[i].name);
    fprintf(fptr, "roll No.: %d\n",stu[i].roll);
    fprintf(fptr, "marks:\n");

    fprintf(fptr, "  C_PROGRAMMING  : %d\n",stu[i].sub.C_PROGRAMMING);   //       |
    fprintf(fptr, "  LINUX          : %d\n",stu[i].sub.LINUX);           //       |
    fprintf(fptr, "  EVS            : %d\n",stu[i].sub.EVS);             //       | --> Marks output
    fprintf(fptr, "  MATHS          : %d\n",stu[i].sub.MATHS);           //       |
    fprintf(fptr, "  PROB_SOLVING   : %d\n",stu[i].sub.PROB_SOLVING);    //       |

    fprintf(fptr, "totalmarks: %d\n",stu[i].totalmarks);
    fprintf(fptr, "percentage: %.2f%%\n",stu[i].percentage);
    fprintf(fptr, "----------------------------\n \n");
    }

    fprintf(fptr,"THANK YOU");

    fclose(fptr);
```

- **Find student record**

```c
printf("do you want to search any student record (y/n): ");
   scanf(" %c",&ques);

   if( ques == 'y' || ques == 'Y'){

       int no;

       printf("enter the number of students you want to see the record of: ");
       scanf("%d",&no);
       for(int i = 0 ; i <=no ; i++){
           int find;

           printf("enter the roll no. of the student you want to see the record of: ");
           scanf("%d",&find);

           int index = searchstudent(stu , num , find);

           if(index != -1){
                printf("name = %s   roll no. = %d    percentage = %.2f%%\n",stu[index].name,stu[index].roll,stu[index].percentage);
           }
           else{
               printf("student not found with that roll number");
           }
       }
   }
```

- **Grading each subject**

```c
int grade(int marks){
    if(marks >= 90) return 10;
    else if(marks >= 80) return 9;
    else if(marks >= 70) return 8;
    else if(marks >= 60) return 7;
    else if(marks >= 50) return 6;
    else if(marks >= 40) return 5;
    else return 0;
}
```

- **GPA**

```c
float calgpa(struct student s){
    int gpa1 = grade(s.sub.C_PROGRAMMING);
    int gpa2 = grade(s.sub.LINUX);
    int gpa3 = grade(s.sub.EVS);
    int gpa4 = grade(s.sub.MATHS);
    int gpa5 = grade(s.sub.PROB_SOLVING);

    return (float)(gpa1+gpa2+gpa3+gpa4+gpa5) / 5.0;
}
```

TESTING AND RESULT

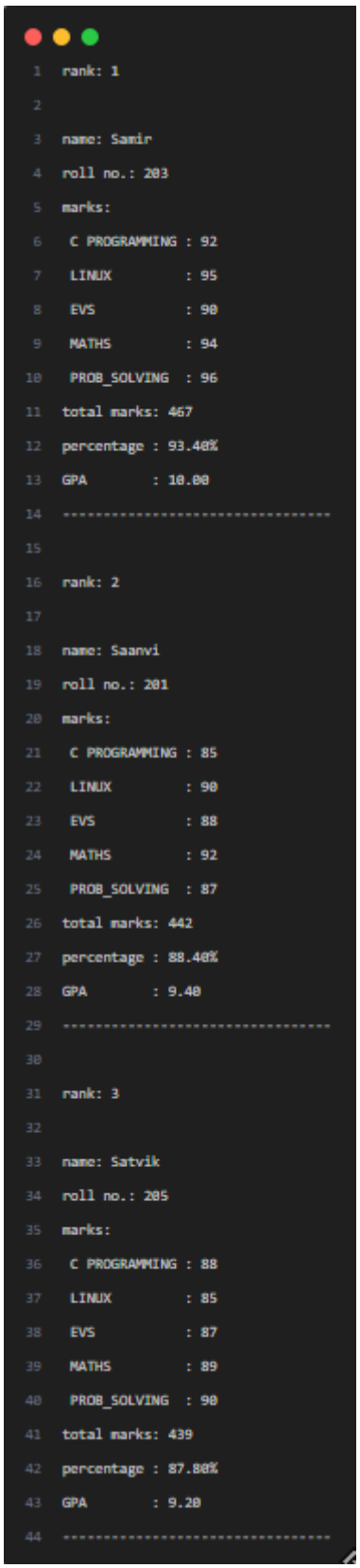
- **Input**

```
Saanvi
201
85 90 88 92 87

Sakshi
202
78 75 80 70 68

Samir
203
92 95 90 94 96

Sanaya
204
65 70 72 68 70

Satvik
205
88 85 87 89 90
```

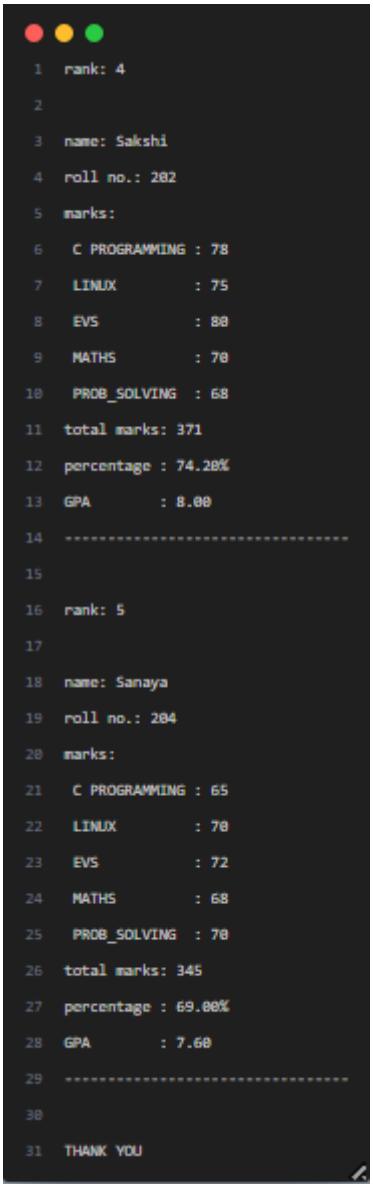
- In this test case, we have entered the information of 5 students, including their names, roll numbers, and marks in all five subjects. This input set is used to demonstrate how the program processes student data, calculates total marks, percentage, and assigns ranks accordingly. The system is fully dynamic, meaning the user can enter any number of students as required

- **Result**
-

- “The following is the final output generated by the Result Management System after processing all student records, calculating their total marks, GPA and percentages, and assigning ranks accordingly.”



```
rank: 1

name: Samir
roll no.: 203
marks:
 C PROGRAMMING : 92
 LINUX         : 95
 EVS           : 90
 MATHS         : 94
 PROB_SOLVING  : 96
total marks: 467
percentage : 93.40%
GPA        : 10.00
--------------------------------

rank: 2

name: Saanvi
roll no.: 201
marks:
 C PROGRAMMING : 85
 LINUX         : 90
 EVS           : 88
 MATHS         : 92
 PROB_SOLVING  : 87
total marks: 442
percentage : 88.40%
GPA        : 9.40
--------------------------------

rank: 3

name: Satvik
roll no.: 205
marks:
 C PROGRAMMING : 88
 LINUX         : 85
 EVS           : 87
 MATHS         : 89
 PROB_SOLVING  : 90
total marks: 439
percentage : 87.80%
GPA        : 9.20
--------------------------------
```



```
rank: 4

name: Sakshi
roll no.: 202
marks:
 C PROGRAMMING : 78
 LINUX         : 75
 EVS           : 80
 MATHS         : 70
 PROB_SOLVING  : 68
total marks: 371
percentage : 74.20%
GPA        : 8.00
--------------------------------

rank: 5

name: Sanaya
roll no.: 204
marks:
 C PROGRAMMING : 65
 LINUX         : 70
 EVS           : 72
 MATHS         : 68
 PROB_SOLVING  : 70
total marks: 345
percentage : 69.00%
GPA        : 7.60
--------------------------------

THANK YOU
```

SEARCH OUPUT

```
do you want to search any student record (y/n): y
enter the number of students you want to see the record of: 1
enter the roll no. of the student you want to see the record of: 203

name: Samir
roll no.: 203
marks:
 C PROGRAMMING : 92
 LINUX         : 95
 EVS           : 90
 MATHS         : 94
 PROB_SOLVING  : 96
total marks: 467
percentage : 93.40%
GPA  :  10.00
rank: 2
--------------------------------
```

When the user chooses to search a student, the program asks for the roll number of the student whose record needs to be viewed.
Using the searchstudent() function, the program locates the student in the structure array.
If the roll number is found, the complete student profile is displayed, including:

- **Student name**
- **Roll number**
- **Individual subject marks**
- **Total marks**
- **Percentage**
- **GPA**
- **Rank based on class performance**

The output is shown in a neatly formatted block for easy readability.

CONCLUSION AND FUTURE WORK

- ***Conclusion***

This project successfully demonstrates how a computerized Result Management System can automate the entire process of handling student academic records. Instead of manually calculating marks, percentages, and ranks—which is time-consuming and prone to human error—the program performs all operations efficiently and with complete accuracy. By taking student details and subject-wise marks as input, the system calculates total marks, percentage, and assigns ranks based on performance.

The program also incorporates searching functionality, allowing users to easily retrieve the result of any student using their roll number. In addition, all generated results are saved in an external file ("RESULT.txt"), ensuring the data is permanently stored and can be accessed even after the program has stopped running. This makes the system practical, reliable, and useful for real-life academic environments.

Overall, the project provides a fast, error-free, and user-friendly approach for managing student results. It reduces manual workload, improves accuracy, and organizes data in a structured manner. This project not only strengthens understanding of structures, file handling, sorting techniques, and modular programming in C, but also demonstrates how automation can simplify complex tasks and increase efficiency.

- ***Future work***

    1. **A simple menu or graphical interface to make it easier to use**
    2. **Saving and loading student data from a file or database**
    3. **Exporting results to PDF or Excel**
    4. **Adding more subjects more classes or attendance details**
    5. **Showing highest marks lowest marks and class average**
    6. **Adding option to edit, update, or delete a student record**

REFERENCES

- **Let Us C**
- **GeeksforGeeks – "C programming"**
- **UPES University course material and class notes**
- **Apna College – "C language tutorial"**
  Link - https://youtu.be/irqbmMNs2Bo?si=qMkQp-gl-soBy-rn

This concludes the project report on

"Result management system (with ranking and GPA)

THANK YOU.