School of Computer and Information Sciences

Software Engineering Lab

Student Result Processing System

Module Specification Document

Submitted by

D. Harshitha Reg. No: 25MCMI16

1. Problem Statement

Design and implement Student Result Processing System that reads student details from a file, validates the input data, computes academic results, assigns grades, and display report of the result.

1.1. Input and Validation

The system should accept details for N students from an input file. For each student, the following data must be provided:

- Student ID: Must be unique and alphanumeric
- Student Name: Should contain only alphabets
- Marks in 5 Subjects
    - Minor Exam: 40 Marks
    - Major Exam: 60 Marks

Validations to be performed:

- Reject duplicate student IDs
- Rejects IDs containing digits or symbols
- Reject names containing digits or symbols
- Reject marks outside the range [0,100]

1.2. Computation

For each valid student record, the system should:

- Calculate Total Marks
- Calculate Percentage
- Minimum passing marks in each subject is 50%

Grade Assignment Based on Percentage:

- ≥ 90 – O
- 85-90 – A+
- 75-85 – A
- 65-75 – B+
- 60-65 – B
- 55-60 – C
- 50-55 – D
- < 50 – F

1.3.Output

The system should display a tabular report containing:

- Student ID
- Student Name
- Subject Marks
- Total Marks
- Percentage
- Grade
- CGPA
- Class Average Percentage
- Highest Percentage
- Lowest Percentage
- Number of students in each grade category

2. Module Specifications

2.1. Module: main()

- **Input:** Opens input and output files.
- **Pre-condition:** Files must exist and contain valid data.
- **Logic:**
  - (i) Open files.
  - (ii) Read student data.
  - (iii) Validate ID, Name, Marks
  - (iv) Compute results.
  - (v) Print results.
- **Output:** Results are written to output file
- **Pseudo-code:**

```
MAIN():
  if argc < 3:
    interactive_mode()
  else:
    open input_file
    open output_file
    while not EOF:
      read student_id, name, marks
      if validation fails:
        print error
        continue
      compute_result(student)
      store student
    print_result(students)
```

2.2. Module: valid_id()

- **Input:** student_id, students[], count
- **Pre-condition:** ID must be 8 characters, alphanumeric

- **Logic:** Check Length, check characters by checking their ascii value, check duplicates by checking if the given ID already exists in the students[].
- **Output:** 0 if valid, 1 if valid
- **Pseudocode:**

```
valid_id(student_id, students, count):
  if length of student_id != 8:
    return 1

  for each character in student_id:
    if character is not alphanumeric:
      return 1

  for i from 0 to count-1:
    if students[i].student_ID == student_id:
      return 1

  return 0
```

2.3. Module: valid_name()

- **Input:** name
- **Pre-condition:** Only alphabetic letters allowed
- **Logic:** Loop through each character, check if it is alphabet by checking their ascii value.
- **Output:** 0 if valid, 1 if invalid
- **Pseudocode:**

```
valid_name(name):
  for each character in name:
    if character is not alphabet:
      return 1
  return 0
```

2.4.Module: valid_marks()

- **Input:** marks[5][2]
- **Pre-condition:** Minor (0-40), Major (0-60)
- **Logic:** Validate each subject marks.
- **Output:** 0 if valid, 1 if invalid
- **Pseudocode:**

```
valid_marks(marks):
  for i from 0 to 4:
    if marks[i][0] < 0 or marks[i][0] > 40:
      return 1
    if marks[i][1] < 0 or marks[i][1] > 60:
        return 1
    return 0
```

2.5. Module: compute_result()

- **Input:** Student *s
- **Pre-condition:** Valid student marks
- **Logic:**
  (i) Sum marks for each subject.
  (ii) If subject total < 50, failed.
  (iii) Calculate total, percentage, grade, CGPA
- **Output:** Updated student structure with total, percentage, grade, CGPA
- **Pseudocode:**

```
compute_result(student):
  student.total = 0

  for i from 0 to 4:
    subject_total = student.marks[i][0] + student.marks[i][1]

    if subject_total < 50:
      student.marks[i][0] = 0
      student.marks[i][1] = 0
      subject_total = 0

    student.total = student.total + subject_total

  student.percentage = (student.total / 500) * 100
  student.cgpa = student.percentage / 10

  if student.percentage >= 90:
    student.grade = "O"
  else if student.percentage >= 85:
    student.grade = "A+"
  else if student.percentage >= 75:
    student.grade = "A"
  else if student.percentage >= 65:
    student.grade = "B+"
  else if student.percentage >= 60:
    student.grade = "B"
  else if student.percentage >= 55:
    student.grade = "C"
  else if student.percentage >= 50:
    student.grade = "D"
  else:
```

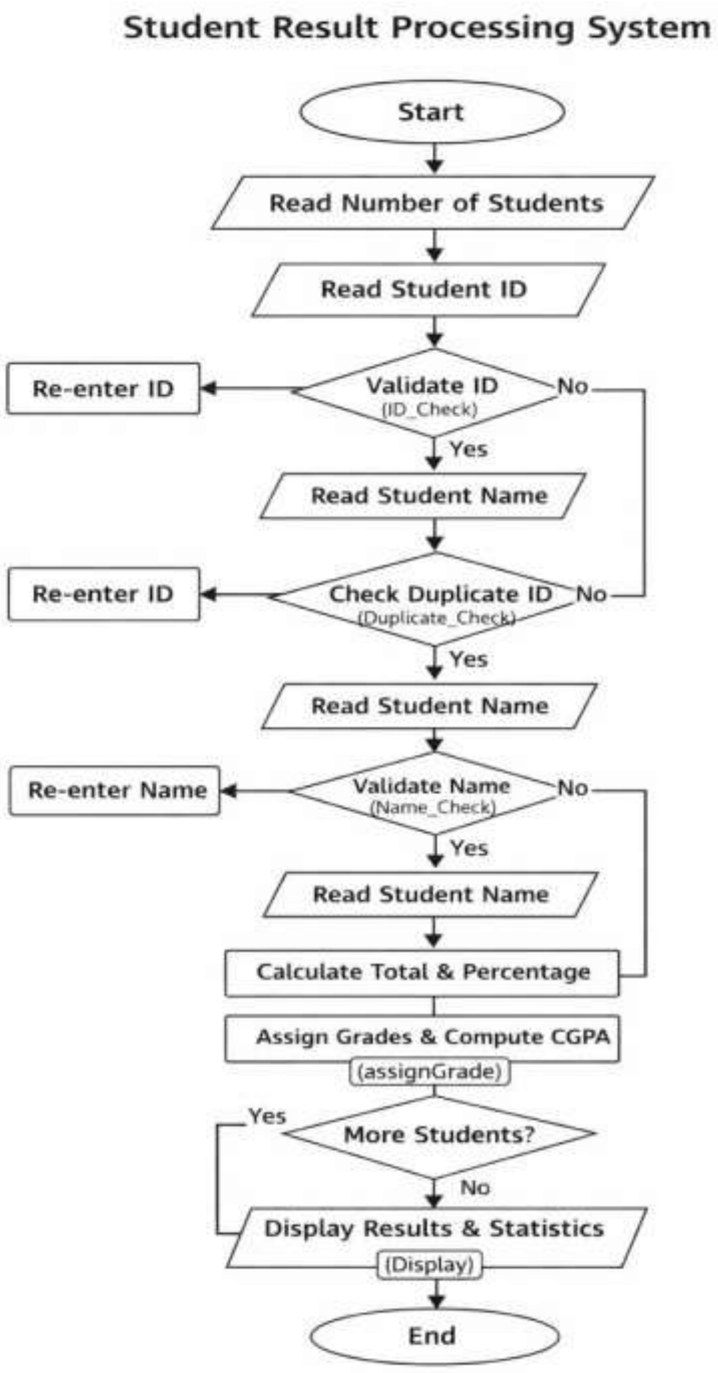
```
student.grade = "F"
```

2.6. Module: print_result()

- **Input: students[], count, FILE *fout**
- **Pre-condition: Results computed and written to the output file**
- **Logic:**
  (i) Calculate class highest, lowest, average
  (ii) Count grades
  (iii) Print class performance
  (iv) Print student-wise details.
- **Output:** Class and student wise results are written to the output file.
- **Pseudocode:**

```
print_result(students, count, output_file):
  highest = students[0].percentage
  lowest = students[0].percentage
  sum = 0
  grade_count[8] = {0}

  for i from 0 to count-1:
    if students[i].percentage > highest:
      highest = students[i].percentage
    if students[i].percentage < lowest:
      lowest = students[i].percentage
    sum = sum + students[i].percentage

    increment grade_count based on students[i].grade

  write "Class Performance" to output_file
  write highest, lowest, sum/count to output_file
  write "Grade Distribution" to output_file

  for i from 0 to count-1:
    write student ID, name to output_file
    for j from 0 to 4:
      write marks[j][0], marks[j][1], sum to output_file
    write total, percentage, CGPA, grade to output_file
```

3. Flowchart



Student Result Processing System
Start
Read Number of Students
Read Student ID
Validate ID
(ID_Check)
No
Yes
Re-enter ID
Read Student Name
Check Duplicate ID
(Duplicate_Check)
No
Yes
Re-enter ID
Read Student Name
Validate Name
(Name_Check)
No
Yes
Re-enter Name
Read Student Name
Calculate Total & Percentage
Assign Grades & Compute CGPA
(assignGrade)
More Students?
Yes
No
Display Results & Statistics
(Display)
End