Methodology of Problem Solving – WGUI

---

1. Understanding the Problem

The first step is to clearly understand what the program must do:

- Input **grades** for students in **two classes** (ITC I21, ITC 122).
- Calculate **final grades** using **weighted formula**:
  Final Grade=(Midterm×0.40)+(Finals×0.60)\text{Final Grade} = (\text{Midterm} \times 0.40) + (\text{Finals} \times 0.60)Final Grade=(Midterm×0.40)+(Finals×0.60)
- Determine **pass/fail** status (final grade ≥ 75 → PASS).
- Track total **passed** and **failed students**.
- Compute **percentage**, **ratio**, and check **class passing rate**.
- Validate **inputs** (numeric, within range).
- Present results in an **easy-to-read GUI table**.

---

2. Problem Decomposition

Break down the problem into smaller, manageable tasks:

Step 1: Class Selection

- Present options: ITC I21 or ITC 122.
- Determine the required passing rate based on the class:
  - ITC I21 → 70%
  - ITC 122 → 40%

Step 2: Set Number of Students

- Input total number of students (0–99).
- Validate input: numeric, within range.
- Initialize counters: `passed`, `failed`, `enteredStudents`.

Step 3: Input Grades

- Loop for each student:
    - Input **Midterm grade**
    - Input **Finals grade**
    - Validate numeric input and range (0–100).
- Calculate **final grade**.
- Determine **status (PASS/FAIL)**.
- Update **counters**.

Step 4: Display Individual Student Info

Show a **table-like format** with:

```
Student | Midterm | Finals | Final | Status
```

-

Step 5: Display Summary

- Total passed and failed students.
- Number of students passed out of total.
- Percentage of students passed and failed.
- Ratio of passed to failed.

- Check if class meets the **required passing rate**.

---

3. Input / Output Identification

Type	Input	Output
Selection	Class (ITC I21 / ITC 122)	Required passing rate (70% / 40%)
Integer	Number of students (0–99)	Initialize counters
Double	Midterm grade (0–100)	Display in table, calculate final grade
Double	Finals grade (0–100)	Display in table, calculate final grade
Computed	Final grade (Midterm *0.4 + Finals*0.6)	Display status (PASSED/FAILED), percentages, ratio

---

4. Process / Algorithm

1. **Select Class** → determine passing rate.
2. **Enter total students** → validate range (0–99).
3. **Initialize counters** → `passed = 0`, `failed = 0`, `enteredStudents = 0`.
4. **Loop for each student**:
   - Input midterm and finals grades (validate numeric, 0–100).
   - Calculate final grade.
   - If final grade ≥ 75 → `passed++`, else → `failed++`.
   - Append student info to table output.
5. **After all students**:
   - Calculate pass % and fail %.

    - Display summary with passed, failed, ratio, and class passing rate check.

---

5. Constraints / Validation

- Number of students: **0–99**
- Grades: **0–100**, numeric only
- Passing grade: **≥ 75**
- Cannot enter grades beyond total number of students
- Class passing rate: ITC I21 = 70%, ITC 122 = 40%

---

6. Edge Cases

- Total students = 0 → percentages = 0%
- Invalid numeric input → prompt user again
- Grades below 0 or above 100 → rejected
- User tries to enter more grades than total students → prevented
- Pass % exactly at threshold → counts as PASSED

---

7. Solution Design / Flow

High-level steps:

1. Start → Display GUI → Select class
2. Set total students → Initialize counters

3. Loop → Input Midterm and Finals → Validate → Compute Final Grade → Update counters → Display student info
4. After all students → Compute % and ratio → Display summary → Check required passing rate
5. End

Optional: Reset GUI to start a new class

---

8. Tools / Techniques Used

- **Java Swing** → GUI interface
- **JTextField, JComboBox, JTextArea** → Input & Output
- **GridLayout / BorderLayout** → organize GUI components
- **ActionListener** → handle button clicks
- **String.format / printf** → display grades & percentages with 2 decimal places
- **Counters** → passed, failed, enteredStudents
- **Input validation** → numeric & range checks