National University of Computer and Emerging Sciences

Fast School of Computing Fall 2024

CS1002-Programming Fundamentals (CS-A,B,C,D,E,F,G)
Assignment 02

Instructions for submission:

Dear students we will be using auto-grading tools, so failure to submit according to the below format would result in zero marks in the relevant evaluation instrument.

i. For each question in your assignment, make a separate cpp file e.g. for question 1, make ROLL-NUM_SECTION_Q#.cpp (23i-0001_A_Q1.cpp) and so on. Each file that you submit must contain your name, student-id, and assignment # on top of the file in comments.
ii. Combine all your work in one folder. The folder must contain only .cpp files (no binaries, no exe files etc.).
iii. Run and test your program on a lab machine before submission.
iv. Rename the folder as ROLL-NUM_SECTION (e.g. 23i-0001_A) and compress the folder as a zip file. (e.g. 23i-0001_A.zip). do not submit .rar file.
v. Submit the .zip file on Google Classroom within the deadline.
vi. Submission other than Google classroom (e.g. email etc.) will not be accepted.
vii. The student is solely responsible to check the final zip files for issues like corrupt file, virus in the file, mistakenly exe sent. If we cannot download the file from Google classroom due to any reason it will lead to zero marks in the assignment.
viii. Displayed output should be well mannered and well presented. Use appropriate comment and indentation in your source code.
ix. Total Marks: 110.
x. If there is a syntax error in code, zero marks will be awarded in that part of assignment.
xi. Your code must be generic.
xii. Solve the assignment by using the concepts of conditional/decision structures as well as the concepts we have studied before that.
xiii. You cannot use advance constructs like loops / arrays for this assignment
xiv. Try to submit your assignment 3 hours before the deadline to avoid any problem(e.g; Internet issue etc)

Deadline:

Deadline to submit assignment is 11th October, 2024 11:59 PM. You are supposed to submit your assignment on GOOGLE CLASSROOM (CLASSROOM TAB not lab). Only ".ZIP" files are acceptable. Other formats should be directly given ZERO. Correct and timely submission of the assignment is the responsibility of every student, hence no relaxation will be given to anyone. **Late Submission policy will be applied as described in course outline**.

Tip: For timely completion of the assignment, start as early as possible.

Plagiarism: Plagiarism is not allowed. If found plagiarized, you will be awarded zero marks in the assignment (copying from the internet is the easiest way to get caught).

Note: Follow the given instruction to the letter, failing to do so will result in a zero.

General Instructions for the assignment:

1. **Variable and Function Naming:** Use variables and function names that reflect the context of the problem. Avoid generic names like `x`, `y`, or `z`.

2. **Logical Thinking:** In your code comments, explain why you chose specific variable names and why a particular operation (like `+` or `%`) is necessary for the problem's solution. These comments will be checked for correctness.

3. **Code Structure:** Your program should contain a clear `main()` function that primarily calls other functions where appropriate. Break down your program into smaller functions where necessary.

4. **Comments and Documentation:** Add a comment at the top of your code that includes your name, roll number, and a brief description of the program. Each function should have a comment explaining its purpose and parameters. Use comments to explain any non-obvious parts of your code.

5. **Input/Output Handling:** Provide clear instructions when taking input from the user. Format your output clearly, ensuring it's easy to understand and follows the requirements of the scenario in the assignment.

Evaluation Criteria

1. Your assignment will be evaluated based on:
2. Correctness: Does the program produce the correct results for all inputs?
3. Complexity: Are multiple conditions and adjustments applied using appropriate decision structures (e.g. switch, nested if-else statements and ternary operators etc)?
4. Efficiency: Is the code clean, efficient, and well-commented?
5. Comprehensive Output: Does the program handle all scenarios with clear and concise output?
6. Error Handling: Ensure the program manages invalid inputs gracefully

Question 1: [30 marks]

Interstellar Trading System: Case Study

Background

It's the year 2150, and interplanetary trade is the lifeblood of the solar system's economy. The Interstellar Trading Consortium (ITC) controls all commerce between Earth, Mars, and Jupiter. As the lead software engineer at ITC, you are tasked with developing an essential part of their trading platform: a program to decide trade approvals, calculate taxes, and determine bonuses for traders based on various complex factors.

However, the trading rules have recently been updated to prevent exploitation and make things more challenging for traders. Your task is to implement these rules in a single C++ program using only conditional structures (if, else), nested conditions, and some ternary operators for good measure. The goal is to create a program that processes trade requests and provides accurate decisions based on highly specific criteria.

The Challenge

Your job is to design and implement a system that processes trade requests for ITC by following these new rules. The program will need to evaluate multiple factors, ranging from trader experience to trade value, market volatility, and even interplanetary politics!

Inputs:

For each trade request, your program should take the following inputs:

Home planet: of the trader (Earth, Mars, or Jupiter)

Experience level: of the trader (Novice, Intermediate, or Expert)

Trade value: (an integer representing credits)

Number of previous successful trades: (an integer)

Rare resources flag: (a boolean indicating if the trade involves rare resources)

Market volatility: (Low, Medium, or High)

Interplanetary trade quota: (percentage value representing how much of their trade quota is used)

Additional Conditions:

These additional factors need to be considered in decision-making:

1. Age of the trader (integer): Traders above 60 years get an additional 2% tax discount due to seniority benefits.

2.Planetary trade sanctions: If Earth is under trade sanctions, no trades can originate from Earth regardless of other conditions.

3.Political alliances: Jupiter traders receive an automatic 5% tax reduction if market volatility is Medium due to a political agreement between Jupiter and the ITC.

Part 1: Trade Eligibility Criteria

The first task is to determine whether the trader is eligible to conduct the trade.

- Traders from **Earth** are always eligible unless there are trade sanctions in place.

- Traders from **Mars** are eligible if they:

- Have at least 10 successful trades, or
- Their trade value exceeds 15,000 credits, and the market volatility is not Low.

- Traders from **Jupiter** are eligible only if:

- They are **Expert traders**,
- The trade involves **rare resources**, and
- Their quota usage is below 80%.

If any of these conditions are not met, the trade should be denied, and the program should clearly state the reason for the denial.

Part 2: Tax Calculation (Nested and Ternary Operations)

Once eligibility is determined, your program must calculate the tax based on a variety of factors. Start with a base tax rate:

- **Earth: 5%**
- **Mars: 8%**
- **Jupiter: 12%**

Apply the following adjustments:

- **+2%** for **Novice** traders.
- **-1%** for **Expert** traders.
- **+3%** if the trade value exceeds 50,000 credits.
- **-2%** if the trader has more than 20 successful trades.
- **-5%** for Jupiter traders if the market volatility is **Medium** due to political alliances.

- **+4%** if market volatility is **High**, regardless of the planet.

Finally, ensure that the tax rate never drops below 0% using a **ternary operator**. Also, check if the trader is over 60 years old; if so, subtract another **2%** from the final tax as a senior discount.

Part 3: Bonus Eligibility and Calculation

Determine whether the trader is eligible for a bonus:

A trade must be approved for a trader to qualify for a bonus.

A trader is eligible for a bonus if:

- The trade involves **rare resources** and market volatility is **High**; or
- The trader is **Expert level**, has over 30 successful trades, and the trade value exceeds 100,000 credits.

Bonus amounts

5% of the trade value for Earth or Mars traders.

8% of the trade value for Jupiter traders who meet the bonus criteria.

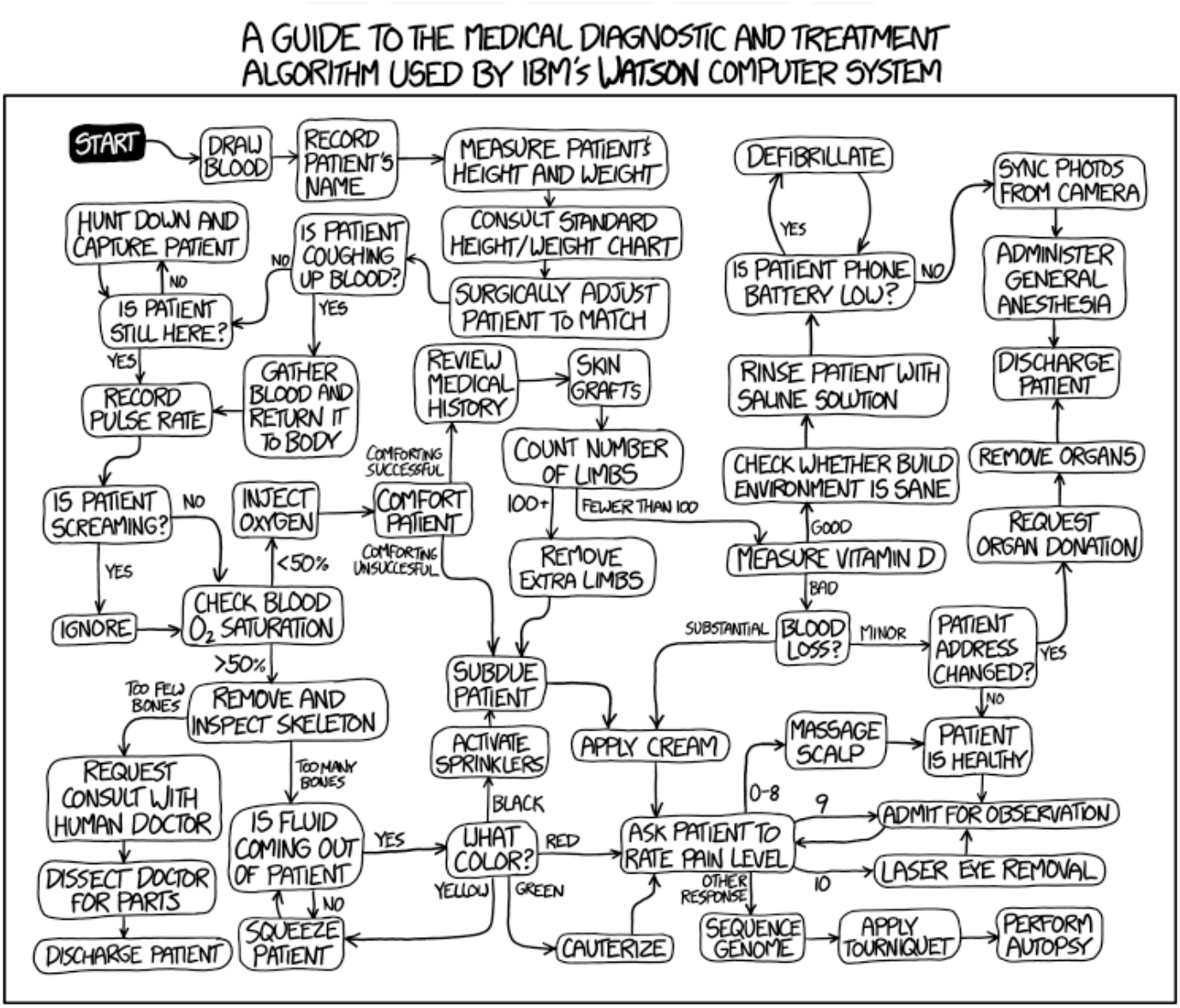
Part 4: Output

Your program should output the following:

- **Trade Status**: Whether the trade is approved or denied.
- If **denied**, the specific reason for denial should be given.
- If **approved**, display:
    1. The **final tax rate**.
    2. The **tax amount** (trade value multiplied by the tax rate).
    3. Whether a **bonus** is awarded, and if so, the **bonus amount**

Question 2:[30] - **IBM Watson's Medical Diagnostic and Treatment Algo**

You are tasked with implementing a **C++ program** that simulates the decision-making process represented in the flowchart provided. Additionally, the flowchart you received is not well-structured using standard flowchart notation, so you are also required to **redesign the flowchart using appropriate symbols** before proceeding with the programming task.



Flowchart Description:

The flowchart simulates a **medical diagnostic and treatment algorithm** used by IBM's Watson computer system. The process involves diagnosing and treating a patient based on various conditions, using decisions related to patient status, medical history, symptoms, and environmental factors. The steps include measuring patient vitals, checking medical conditions, treating the patient based on the diagnosis, and handling emergencies.

However, the current flowchart uses non-standard symbols and has humorous elements that don't reflect professional medical decision-making processes accurately. Your job is to first **correct the flowchart**, using proper notation to represent decision points, processes, inputs, and outputs clearly.

Instructions:

1. **Step 1: Redraw the Flowchart (hand-drawn)**:
    - Use standard flowchart symbols to represent decision points (diamond), actions/processes (rectangle), and input/output (parallelogram).
    - Correctly represent the flow of logic from the start to the termination points, keeping the key elements of the original flowchart intact. Ensure all steps are logically connected.
    - Remove unnecessary comedic elements and add clarity to each step, using meaningful labels for processes and decisions (e.g., replacing "surgically adjust patient" with "adjust patient's posture").
    - Submit your new, neatly drawn flowchart using appropriate flowchart notation. (Make sure you submit a hand drawn flowchart by writing your name and roll number on the top. You can scan and submit the flowchart alongwith the code file).
2. **Step 2: Implement a C++ Program**:
    - Once the flowchart is redrawn, write a **C++ program** that follows the logical structure of the redesigned flowchart.
    - The program should simulate the diagnostic steps, allowing users to input patient conditions (e.g., coughing blood, screaming, pulse rate, etc.) and providing corresponding treatment steps based on the user's input.
    - Use appriopriate decision structures **(e.g. switch statements , if, if else or nested if-else etc)** to handle the different decision points.
    - For processes like "draw blood" or "measure height and weight," simulate these actions using simple print statements (e.g., "Drawing blood...").
3. **Program Flow**:
    - The program starts by simulating the initial checks (e.g., is the patient present? Is the patient screaming?).
    - Depending on the user inputs, the program follows through different paths (e.g., subduing the patient, applying cream, removing organs) based on the decisions from the flowchart.
    - Use **appropriate function calls** for each major action (you do not need to implement the full medical treatment, just simulate it using function calls like applyCream(), comfortPatient(), etc.).

Example Inputs and Outputs:

Sample Input:

1. Is patient still here? (yes/no): yes

2. Is patient coughing up blood? (yes/no): no

3. Is patient screaming? (yes/no): yes

4. Oxygen saturation <50%? (yes/no): yes

Sample Output:

Patient is still here.

Patient is not coughing blood.

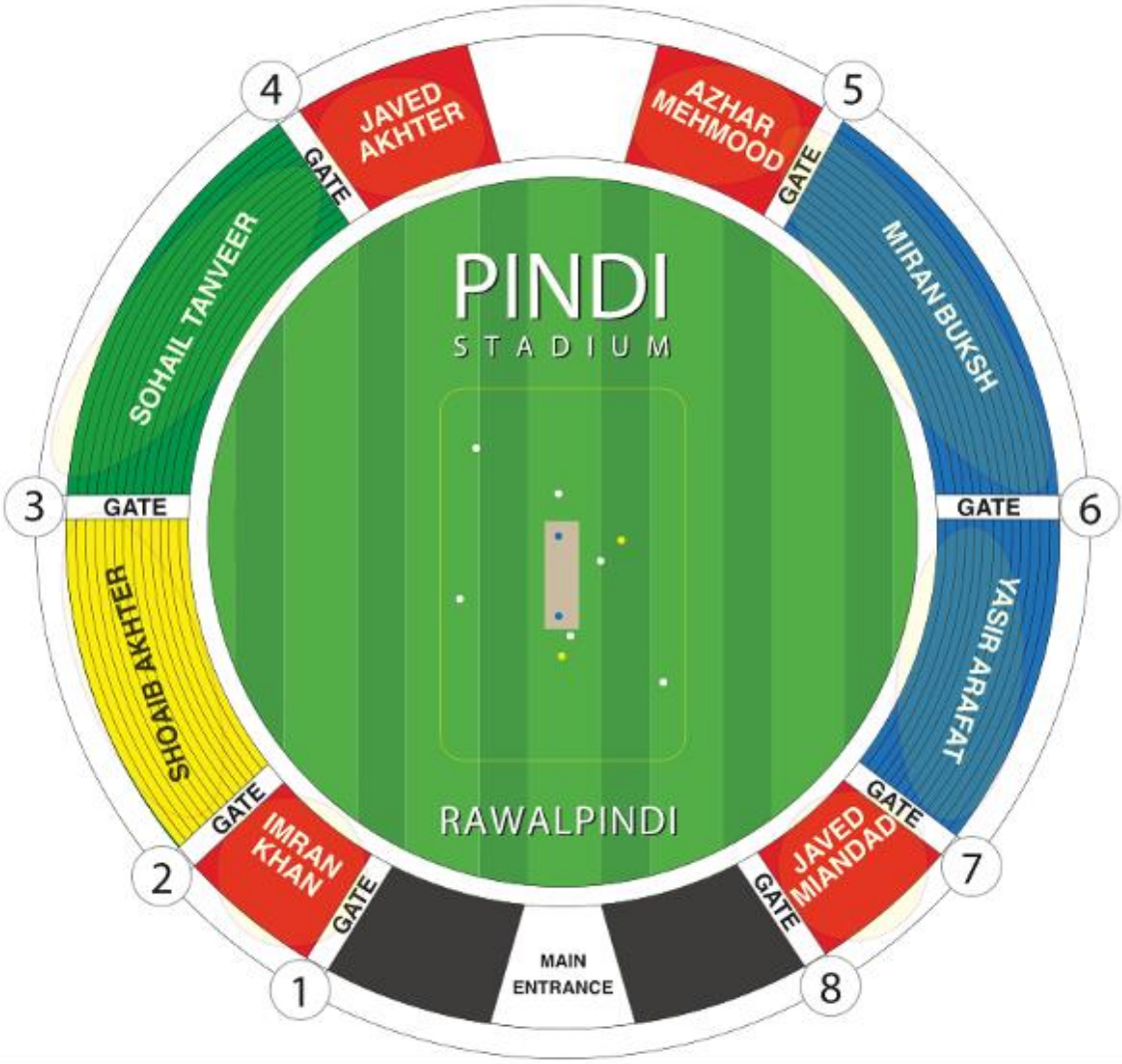
Patient is screaming.

Oxygen saturation is below 50%. Inject oxygen. Checking blood oxygen saturation...

Discharge patient.

Question 3:[30] - **PSL-09 E-Ticketing System (2024) - Rawalpindi Cricket Stadium**

Write a **C++ program** for a company that is planning to sell **e-tickets** for the PSL-09 matches being held at **Rawalpindi Cricket Stadium (RCS)** in 2024. The program should simulate the ticket-buying process, including enclosures, pricing, available seats, and discount calculations.



Stadium Seating Enclosures:

The stadium has the following enclosures (as also shown in the above figure), each with a **maximum seating capacity as follows**:

- **VIP**: Black ones are VIP enclosures with 250 seating capacity each.
- **Premium**: Red ones are Premium enclosures with 1000 seating capacity each.

- **First-class**: Yellow and Green ones are first class enclosures with 2000 seating capacity each.
- **General**: Blue ones are General enclosures with 2500 seating capacity each.

  In your code you cannot use Black, Red etc for the enclosures - instead use their appropriate name for each of the enclosure in your program.

Ticket Prices (Dynamic Based on Match Type):

Prices differ for day and night matches as follows:

1. **Night Matches** (7 pm - 10:15 pm):
   - **VIP**: Rs. 3000
   - **Premium**: Rs. 1500
   - **First-class**: Rs. 1000
   - **General**: Rs. 500
2. **Day Matches** (2 pm - 5:15 pm):
   - **VIP**: Rs. 2000
   - **Premium**: Rs. 1000
   - **First-class**: Rs. 500
   - **General**: Rs. 250

Match Schedule:

1. **27 Feb 2024** – Islamabad United v Quetta Gladiators (Night Match)
2. **28 Feb 2024** – Peshawar Zalmi v Lahore Qalandars (Night Match)
3. **29 Feb 2024** – Islamabad United v Peshawar Zalmi (Night Match)
4. **1 Mar 2024** – Islamabad United v Karachi Kings (Night Match)
5. **2 Mar 2024** – Peshawar Zalmi v Karachi Kings (Night Match)
6. **5 Mar 2024** – Peshawar Zalmi v Quetta Gladiators (Night Match)
7. **7 Mar 2024** – Peshawar Zalmi v Islamabad United (Day Match)
8. **8 Mar 2024** – Multan Sultans v Islamabad United (Day Match)

Program Flow:

1. **Menu Display**:
   - Display a menu listing the schedule of matches.
   - Allow the user to select a match (1 - 8).
2. **Enclosure and Ticket Selection**:
   - After selecting a match, display a sub-menu showing the names of all enclosures, ticket prices (based on day or night match), and randomly generated available seats for each enclosure (within their capacity limits).
   - The user can select an enclosure and proceed to purchase tickets.
3. **Family and Group Ticket Option**:
   - If the user selects an enclosure, prompt them to choose between buying **Family Tickets** or **Individual Tickets**.
     - **Family Tickets**: Ask for the number of **adult** and **children** tickets (up to **4 adults** and **5 children**), along with their **CNIC**.
     - **Individual Tickets**: Ask for the number of **adult** tickets (up to **8 adults**) and their **CNIC**.
4. **Dynamic Discount System**: Discounts are applied based on the type of tickets, the number of tickets purchased, and the selected enclosure.
   - **VIP Enclosure**:

        - No discount unless booking more than 5 tickets (5% for adults and children).
        - Additional 2% discount for every adult ticket over 5 (up to a max of 15%).
    - **Premium Enclosure**:
        - 10% discount on all family bookings.
        - 20% rebate on children's tickets.
        - Groups of more than 4 adults get a 12% discount.
    - **First-Class and General Enclosure**:
        - Discounts vary with the number of tickets:
            - 3-5 tickets: 5% discount.
            - 6-8 tickets: 10% discount.
            - More than 8 tickets: 15% discount.
    - **Progressive Rebates**:
        - If the total number of tickets (adults + children) exceeds 6, apply an additional 5% discount on the total amount.
        - Children under 5 get a 30% discount on their tickets, and children between 5-10 years get a 25% discount.
5. **Additional Conditions**:
    - **High-Demand Matches**: For popular matches (27 Feb 2024, 29 Feb 2024), a **10% surcharge** is applied to all ticket prices.
    - **Day Matches**: Offer an additional 5% discount if the total number of tickets exceeds 8.
6. **Input Validation**:
    - Ensure valid inputs:
        - The number of tickets cannot exceed the available seats in the selected enclosure.
        - The number of children must always be greater than 1 when selecting the family option.
        - The number of people must always be greater than 0, and users cannot enter more than the maximum allowed tickets (4 adults and 5 children for family, 8 adults for individual).
7. **Total Cost Calculation**:
    - After all inputs and discounts are applied, calculate and display the **total amount due**.
    - Show a summary of the number of tickets purchased, ticket type, CNIC, selected match, and enclosure.

Example Scenario:

- A user selects **Match 1** (Islamabad United v Quetta Gladiators, 27 Feb 2024).
- The program displays available enclosures and ticket prices.
- The user selects the **Javed Miandad Enclosure** (VIP), with 850 seats available.
- The user opts for **Family Tickets** and enters:
    - 3 adults, 2 children.
    - CNIC: 12345-6789012-3.
- The program applies the following discounts:
    - VIP Enclosure: 10% discount for family bookings.
    - 20% discount for children's tickets.
    - Total amount is calculated and displayed.

Question 4:[20]

Part1

Write a program that accepts a date in the format day/month/year (e.g. 29/01/2024) by accepting the day, month, & year as separate integer inputs. The program should perform the following tasks.

1. **Input Validation:** Verify whether the entered date is valid.
2. **Date Conversion:** If the date is valid, convert it to long date format(29th January, 2024).
3. **Leap Year Check:** Check whether the entered year is a leap year or not.

Part2

Write a program that determines whether two triangles are congruent. The program should also identify and print the type of each triangle (equilateral, isosceles, or scalene) based on its sides.You are required to perform the following tasks: (By using conditional structures)

1. **Input Data:** Accept the lengths of all three sides and the measures of all three angles foreach triangle as inputs.
2. **Congruency Check:** Determine whether the two triangles are congruent by applying congruency criteria (SSS, SAS, ASA, AAS, or RHS).
3. **Triangle Type Identification:** Classify each triangle as equilateral, isosceles, or scalenebased on its sides.
4. **Output Results:** Display whether the two triangles are congruent and print the type ofeach triangle.

Note: Ensure that when checking for congruency, you match the **corresponding sides and angles** of both triangles accurately. This means that side 1 of triangle 1 must be compared to thecorresponding side of triangle 2, and similarly for the angles. Proper alignment of correspondingelements is crucial for an accurate congruency check.

You can visit this link to understand the concept of congruent triangles:

https://www.onlinemathlearning.com/prove-triangles-congruent.htmls

*****___*** HAPPY CODING! ***___*****