Programming Fundamental

Muhammad Atif
241929

Question no 1:

In a right triangle, the square of the length of one side is equal to the sum of the squares of the lengths of the other two sides. Write a program that prompts the user to enter the lengths of three sides of a triangle and then outputs a message indicating whether the triangle is a right triangle.

Explanation:

Declaring variables:

- Variables a, b, and c are declared to represent the three sides of a triangle. Variables hyp, side1, and side2 are declared to hold the hypotenuse with the two sides.

Input from User:

- The program demands input from the user to enter values representing the three sides of the triangle.

Identify the Hypotenuse:

- The largest of the three values entered is assigned to hyp as that represents the hypotenuse.
- The other two are assigned names side1 and side2. This is achieved using if-else statements when the values of a, b, and c are compared.

Check if the Triangle is a Right-Angle Triangle:

- The Pythagorean theorem is applied:
- A triangle is a right-angle triangle if:

(hypotenuse)^2 = (side1)^2 + (side2)^2

- This condition is checked using an if statement.
- If the condition is true, its display The Triangle is Right Angle Triangle otherwise it display the Triangle is not Right Angle Triangle.

End of program:

- The program returns 0 to indicate that it has successfully completed its execution.

Question no 2:

A box of cookies can hold 24 cookies, and a container can hold 75 boxes of cookies. Write a program that prompts the user to enter the total number of cookies, the number of cookies in a box, and the number of cookie boxes in a container. The program then outputs the number of boxes and the number of containers to ship the cookies. Note that each box must contain the specified number of cookies, and each container must contain the specified number of boxes.If the last box of cookies contains less than the number of specified cookies, you can discard it and output the number of leftover cookies. Similarly, if the last container contains less than the number of specified boxes, you can discard it and output the number of leftover boxes.

Explanation:

Declaring Variable:

We need to declare the following variables:

- totalcookies: Stores the **total number of cookies** entered by the user.
- cookiesperbox: Stores the **number of cookies that fit in one box**.
- boxespercontainers: Stores the **number of boxes per container**.
- totalboxes: Stores the **calculated total number of boxes** required.
- leftovercookies: Stores the **number of leftover cookies** after boxing.
- Totalcontainers:: Stores the **total number of containers** needed.
- leftoverboxes: Stores the **leftover boxes** that could not fill a container.

Input total number of containers:

- It prompts the user to enter total cookies of cookies. This number is stored in the variable totalcookies.

Validate Cookies per Box Input:

- Implement a do-while loop in such a way that the user shall be prompted until he accepts the correct number of cookies in the box.
- If the user input exceeds 24 cookies, then the program prints an alert message.
- The program reminds the user that the minimum number of cookies allowed per box is 1 if he inputs 0 or fewer cookies. The loop repeats until a valid number of cookies between 1 and 24 is input.

Prompt User for Cookies per Box:

- The number of boxes per container is asked through a do-while loop.

- If the user input more than 75 boxes, a reminder is given .It also informs them that if they submit 0 or fewer boxes, the minimum is 1. It then keeps repeating the loop until an integer between 1 to 75 inclusive is actually entered.

Calculate Boxes and Containers:

- totalboxes: The total number of boxes required is calculated by dividing the total cookies by cookies per box.
- leftovercookies: The leftover cookies are calculated using the **modulus operator** %.
- totalcontainers: The total number of containers required is calculated by dividing the total boxes by boxes per container.
- leftoverboxes: The leftover boxes are calculated using the **modulus operator** %.

Display output:

The program output the following information:

- Total number of boxes required.
- Number of leftover cookies.
- Total number of containers required.
- Number of leftover boxes.

End of program:

- The program returns 0 to indicate that it has successfully completed its execution.

Question no 3:

Write a C++ program that takes input from the file in the form of seconds and converts it into a comprehensive time representation. The program should take input from the file having the number of seconds, perform calculations to convert these seconds into years, months, days, hours, minutes, and seconds, and display the results. Furthermore, it should offer the user the option to convert the seconds into weeks, days, hours, minutes, and seconds. The challenge lies in handling various time units accurately, considering leap years, and dealing with months of different lengths, all while providing clear and user-friendly output. This program should ensure both accuracy and user-friendliness in its time conversions. Note: File contains more than one number. Your program should convert each number accordingly.

Explanation:

- First we initializing Constant variables for seconds in different time units.
- declaration of the file that contains the data for seconds.
- Then we check whether the file exists and is readable .
- initializing the variables that will be used for displaying the result.
- Reading and performing calculations on multiple values from the file.

- Reading the seconds from the file and storing it in the second variable.
- calculating number of years from seconds. Considering the possibility of leap years
- calculating number of months from remaining seconds. Calculating number of months for first 7 months
- find out if it is the Feb month of leap year or normal year. Then we calculating for 30 and 31 days long months.
- calculating number of days from remaining seconds, calculating number of hours from remaining seconds, calculating number of minutes from remaining seconds. After that we will display the result in desired format that we use in .cpp file.
- Asking the user for conversion to weeks, days, hours, minutes and seconds if required. Using if condition. If user says 'y' ,then seconds will be converted into weeks, days, hours, minutes, and seconds.
- Displaying the output and close the file.

Question no 4:

Explanation:

- Declares the variables to hold coefficients of the quadratic equation (a, b, c), the roots of the equation (root, root1, root2), and the discriminant (discriminant).
- Prompts the user to input the coefficient 'a' and stores the input in variable a. similarly Prompts the user to input the coefficient 'b 'and stores the input in variable b, and in last Prompts the user to input the coefficient 'c' and stores the input in variable c.
- Prompts the user to input the coefficient 'a' and stores the input in variable a. similarly Prompts the user to input the coefficient 'b 'and stores the input in variable b, and in last Prompts the user to input the coefficient 'c' and stores the input in variable c.
- Checks if 'a' is zero, which would make the equation not quadratic. If it is zero, it prints an error message and exits the program.
- Calculates the discriminant D= b^2 – 4ac, which helps in determining the nature of the roots of the quadratic equation.
- If the discriminant is positive, it calculates two distinct real roots using the quadratic formula and prints them.
- If the discriminant is zero, it indicates one repeated root, which is calculated and printed.
- If the discriminant is negative, it calculates the real and imaginary parts of the roots and prints them, indicating the roots are complex.
- After that output will be displayed.
