1. Write a C program to find maximum between two numbers.

   Helper:

   - Input two numbers from user. Store it in some variable say num1 and num2.
   - Check if(num1 > num2) then print num1 is maximum.
   - Check if(num2 > num1) then print num2 is maximum.
   - Check if(num1 == num2) then both the numbers are equal.

2. Write a C program to find maximum between three numbers.

   Helper:

   1. Input three numbers from user. Store it in some variable say num1, num2 and num3.
   2. Compare first two numbers i.e. num1 > num2. If the statement is true then num2 is surely not max value. Perform one more comparison between num1 with num3 i.e. if(num1 > num3), then num1 is max otherwise num3.
   3. If the statement num1 > num2 is false. Which indicates that num1 is not max. Hence, this time compare num2 with num3. If the statement num2 > num3 is true then num2 is max otherwise num3.

3. Write a C program to check whether a number is negative, positive or zero.

   Helper:

   1. Input a number from user in some variable say num.
   2. Check if(num < 0), then number is negative.
   3. Check if(num > 0), then number is positive.
   4. Check if(num == 0), then number is zero.

4. Write a C program to check whether a number is divisible by 5 and 11 or not.

   Helper:

   1. Input a number from user. Store it in some variable say num.
   2. To check divisibility with 5, check if(num % 5 == 0) then num is divisible by 5.
   3. To check divisibility with 11, check if(num % 11 == 0) then num is divisible by 11.
   4. Now combine the above two conditions using logical AND operator &&. To check divisibility with 5 and 11 both, check if((num % 5 == 0) && (num % 11 == 0)), then number is divisible by both 5 and 11.

5. Write a C program to check whether a number is even or odd.

   Helper:

   1. Input a number from user. Store it in some variable say num.
   2. Check if number modulo division equal to 0 or not i.e. if(num % 2 == 0) then the number is even otherwise odd.

6. Write a C program to check whether a year is leap year or not.

   Helper:

   1. Input year from user. Store it in some variable say year.
   2. If year is exactly divisible by 4 and not divisible by 100, then it is leap year. Or if year is exactly divisible by 400 then it is leap year.

7. Write a C program to check whether a character is alphabet or not.

   Helper:

   1. Input a character from user. Store it in some variable say ch.
   2. Check if((ch >= 'a') && (ch <= 'z')) or if((ch >= 'A') && (ch <= 'Z')). Then it is alphabet otherwise not.

8. Write a C program to input any alphabet and check whether it is vowel or consonant.

   Helper:

   1. Input a character from user. Store it in some variable say ch.
   2. Check conditions for vowel i.e. if(ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u'), then it is vowel.
   3. If character is alphabet but not vowel then it is consonant. Means check ch >= 'a' && ch <= 'z' then, it is consonant.
   4. If it is neither vowel nor consonant, then it is not alphabet.

9. Write a C program to input any character and check whether it is alphabet, digit or special character.

   Helper:

   - A character is alphabet if it in between a-z or A-Z.
   - A character is digit if it is in between 0-9.
   - A character is special symbol character if it neither alphabet nor digit.

   Step by step descriptive logic to check alphabet, digit or special character.

   1. Input a character from user. Store it in some variable say ch.
   2. First check if character is alphabet or not. A character is alphabet if((ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z')).
   3. Next, check condition for digits. A character is digit if(ch >= '0' && ch <= '9').
   4. Finally, if a character is neither alphabet nor digit, then character is a special character.

10. Write a C program to check whether a character is uppercase or lowercase alphabet.

    Helper:

    1. Input a character from user. Store it in some variable say ch.
    2. Character is uppercase alphabet if(ch >= 'A' and ch <= 'Z').
    3. Character is lowercase alphabet if(ch >= 'a' and ch <= 'z').
    4. If none of the above conditions met, then character is not alphabet.

11. Write a C program to input week number and print week day.

    Helper:

    1. Input week day number from user. Store it in some variable say week.
    2. Print Monday if(week == 1). I have assumed Monday as first day of week.
    3. Similarly, check condition for all 7 days and print the corresponding day name.

12. Write a C program to input month number and print number of days in that month.

Helper:

1. Input month number from user. Store it in some variable say month.
2. For each month check separately and print corresponding number of days in that month using above table. For example, print 31 days if month == 1 since, January contains 31 days.
3. Repeat the above step for all 12 months.

13. Write a C program to count total number of notes in given amount.

Helper:

1. Input amount from user. Store it in some variable say amt.
2. If amount is greater than 500 then, divide amount by 500 to get maximum 500 notes required. Store the division result in some variable say note500 = amt / 500;.

   After division, subtract the resultant amount of 500 notes from original amount. Perform amt = amt - (note500 * 500).

3. Repeat above step, for each note 200, 100, 50, 20, 10, 5, 2 and 1.

14. Write a C program to input angles of a triangle and check whether triangle is valid or not.

Helper:

1. Input all three angles of triangle in some variable say angle1, angle2 and angle3.
2. Find sum of all three angles, store sum in some variable say sum = angle1 + angle2 + angle3.
3. Check if(sum == 180) then, triangle can be formed otherwise not. In addition, make sure angles are greater than 0 i.e. check condition for angles if(angle1 != 0 && angle2 != 0 && angle3 != 0).

15. Write a C program to input all sides of a triangle and check whether triangle is valid or not.

Helper:

1. Input sides of a triangle from user. Store them in some variable say side1, side2 and side1.
2. Given triangle is valid if side1 + side2 > side3 and side1 + side3 > side2 and side2 + side3 > side1.

16. Write a C program to check whether the triangle is equilateral, isosceles or scalene triangle.

Helper:

Properties of triangle

- A triangle is said Equilateral Triangle, if all its sides are equal. If a, b, c are three sides of triangle. Then, the triangle is equilateral only if a == b == c.
- A triangle is said Isosceles Triangle, if its two sides are equal. If a, b, c are three sides of triangle. Then, the triangle is isosceles if either a == b or a == c or b == c.
- A triangle is said Scalene Triangle, if none of its sides are equal.

Logic to check equilateral, scalene or isosceles triangle

Step by step descriptive logic to classify triangle as equilateral, scalene or isosceles triangle.

1. Input sides of a triangle from user. Store it in some variables say side1, side2 and side3.
2. Check if(side1 == side2 && side2 == side3), then the triangle is equilateral.
3. If it is not an equilateral triangle then it may be isosceles. Check if(side1 == side2 || side1 == side3 || side2 == side3), then triangle is isosceles.
4. If it is neither equilateral nor isosceles then it scalene triangle.

17. Write a C program to find all roots of a quadratic equation.

Helper:

Quadratic equation

Wikipedia states, in elementary algebra a quadratic equation is an equation in the form of

$$ax^2 + bx + c = 0$$

Solving quadratic equation

A quadratic equation can have either one or two distinct real or complex roots depending upon nature of discriminant of the equation. Where discriminant of the quadratic equation is given by

$$\Delta = b^2 - 4ac$$

Depending upon the nature of the discriminant, formula for finding roots is be given as.

- Case 1: If **discriminant is positive**. Then there are two real distinct roots given by.

$$\frac{-b+\sqrt{\Delta}}{2a} \quad \text{and} \quad \frac{-b-\sqrt{\Delta}}{2a}$$

- Case 2: If **discriminant is zero** then, it has exactly one real root given by.

$$-\frac{b}{2a}$$

- Case 3: If **discriminant is negative** then, it has two distinct complex roots given by.

$$\frac{-b}{2a}+i\,\frac{\sqrt{-\Delta}}{2a} \quad \text{and} \quad \frac{-b}{2a}-i\,\frac{\sqrt{-\Delta}}{2a}$$

Logic to find all roots of a quadratic equation

Based on the above formula let us write step by step descriptive logic to find roots of a quadratic equation.

1. Input coefficients of quadratic equation from user. Store it in some variable say a, b and c.
2. Find discriminant of the given equation, using formula discriminant = (b*b) - (4*a*c).

   Learn - Program to find power of a number.
3. Compute roots based on the nature of discriminant.
4. If discriminant > 0 then, root1 = (-b + sqrt(discriminant)) / (2*a) and root2 = (-b - sqrt(discriminant)) / (2*a).

   Learn - Program to find square root of a number using sqrt() function.
5. If discriminant == 0 then, root1 = root2 = -b / (2*a).
6. Else if discriminant < 0 then, there are two distinct complex roots where root1 = -b / (2*a) and root2 = -b / (2*a).

   Imaginary part of the root is given by imaginary = sqrt(-discriminant) / (2*a).

18. Write a C program to calculate profit or loss.

Helper:

Logic to find profit or loss

In primary mathematics classes, you all have learned about profit and loss. If cost price is greater than selling price then there is a loss otherwise profit.

Formula to calculate profit and loss
Profit = S.P - C.P (Where S.P is Selling Price and C.P is Cost Price)
Loss = C.P - S.P

19. Write a C program to input marks of five subjects Physics, Chemistry, Biology, Mathematics and Computer. Calculate percentage and grade according to following:
Percentage >= 90% : Grade A
Percentage >= 80% : Grade B
Percentage >= 70% : Grade C
Percentage >= 60% : Grade D
Percentage >= 40% : Grade E
Percentage < 40% : Grade F

Helper:

Logic to calculate percentage and grade

In primary mathematics classes you have learned about percentage. Just to give a quick recap, below is the formula to calculate percentage.

$$\textit{Percentage} = \frac{\textit{part}}{\textit{whole}} \times 100$$

Step by step descriptive logic to find percentage and grade.

1. Input marks of five subjects in some variable say phy, chem, bio, math and comp.
2. Calculate percentage using formula per = (phy + chem + bio + math + comp) / 5.0;. Carefully notice I have divided sum with 5.0, instead of 5 to avoid integer division.
3. On the basis of per find grade of the student.
4. Check if(per >= 90) then, print "Grade A".
5. If per is not more than 90, then check remaining conditions mentioned and print grade.

20. Write a C program to input basic salary of an employee and calculate its Gross salary according to following:
Basic Salary <= 10000 : HRA = 20%, DA = 80%
Basic Salary <= 20000 : HRA = 25%, DA = 90%
Basic Salary > 20000 : HRA = 30%, DA = 95%

Helper:

Logic to find gross salary of an employee

Gross salary is the final salary computed after the additions of *DA*, *HRA* and *other allowances*. The formula for *DA* and *HRA* is

da = basic_salary * (DA/100)
If DA = 80% then the statement becomes da = basic_salary * (80/100). Which can also be written as DA = basic_salary * 0.08. Likewise you can also derive a formula for *HRA*.

Step by step descriptive logic to find gross salary of an employee.

1. Input basic salary of employee. Store it in some variable say basic_salary.
2. If basic_salary <= 10000 then, hra = basic_salary * 0.8 and da = basic_salary * 0.2.
3. Similarly check basic salary and compute hra and da accordingly.
4. Calculate final gross salary using formula gross_salary = basic_salary + da + hra.

21. Write a C program to input electricity unit charges and calculate total electricity bill according to the given condition:
For first 50 units Rs. 0.50/unit
For next 100 units Rs. 0.75/unit
For next 100 units Rs. 1.20/unit
For unit above 250 Rs. 1.50/unit
An additional surcharge of 20% is added to the bill

Helper:

Logic to calculate net electricity bill

Step by step descriptive logic to compute electricity bill.

1. Input unit consumed by customer in some variable say unit.
2. If unit consumed less or equal to 50 units. Then amt = unit * 0.50.
3. If unit consumed more than 50 units but less than 100 units. Then add the first 50 units amount i.e. 25 to final amount and compute the rest 50 units amount. Which is given by amt = 25 + (unit-50) * 0.75. I have used units-50, since I already calculated first 50 units which is 25.
4. Similarly check rest of the conditions and calculate total amount.
5. After calculating total amount. Calculate the surcharge amount i.e. sur_charge = total_amt * 0.20. Add surcharge amount to net amount. Which is given by net_amt = total_amt + sur_charge.