Programming Exercises

1. Write programs for the following.

   (a) Computer area of circle, rectangle, triangle and square.

   (b) Compute simple Interest, Compound interest.

   (c) Enter two integers and perform all mathematical operations on it.

   (d) Enter five floating point numbers and display its average.

2. Write program to display the salary sheet of an employee when DA is give @80% of BP, HRA @40% of (BP + DA), CA as Rs. 4750.00. PF is deducted @12.5% of Gross Pay and income Tax deducted @8.2% of Gross Pay. Display DA, HRA, CA, Gross Pay, Income Tax, PF and Net Pay.

3. A Shop keeper gives 10.2% discount on Gross amount, Enter unit rate, quantity purchased calculate Gross amount, discount amount and Net amount and display.

4.
5.

1. Write a program that converts time in minutes to time in hours and minutes. Use #define or const to create a symbolic constant for 60. Use a while loop to allow the user to enter values repeatedly and terminate the loop if a value for the time of 0 or less is entered.

2. Write a program that asks for an integer and then prints all the integers from (and including) that value up to (and including) a value larger by 10. (That is, if the input is 5, the output runs from 5 to 15.) Be sure to separate each output value by a space or tab or newline.

3. Write a program that asks the user to enter the number of days and then converts that value to weeks and days. For example, it would convert 18 days to 2 weeks, 4 days. Display results in the following format:

18 days are 2 weeks, 4 days.

Use a while loop to allow the user to repeatedly enter day values; terminate the loop when the user enters a nonpositive value, such as 0 or -20 .

4. Write a program that asks the user to enter a height in centimeters and then displays the height in centimeters and in feet and inches. Fractional centimeters and inches should be allowed, and the program should allow the user to continue

entering heights until a nonpositive value is entered. A sample run should look like this:

Enter a height in centimeters: **182**

182.0 cm = 5 feet, 11.7 inches

Enter a height in centimeters (<=0 to quit): **168.7**

168.0 cm = 5 feet, 6.4 inches

Enter a height in centimeters (<=0 to quit): **0**

bye

5. Change the program addemup.c (Listing 5.13), which found the sum of the first 20 integers. (If you prefer, you can think of addemup.c as a program that calculates how much money you get in 20 days if you receive $1 the first day, $2 the second day, $3 the third day, and so on.) Modify the program so that you can tell it interactively how far the calculation should proceed. That is, replace the 20 with a variable that is read in.

6. Now modify the program of Programming Exercise 5 so that it computes the sum of the squares of the integers. (If you prefer, how much money you receive if you get $1 the first day, $4 the second day, $9 the third day, and so on. This looks like a much better deal!) C doesn't have a squaring function, but you can use the fact that the square of n is n * n .

7. Write a program that requests a type double number and prints the value of the number cubed. Use a function of your own design to cube the value and print it. The main() program should pass the entered value to this function.

8. Write a program that displays the results of applying the modulus operation. The user should first enter an integer to be used as the second operand, which will then remain unchanged. Then the user enters the numbers for which the modulus will be computed, terminating the process by entering 0 or less. A sample run should look like this:

This program computes moduli.

Enter an integer to serve as the second operand: **256**

Now enter the first operand: **438**

438 % 256 is 182

Enter next number for first operand (<= 0 to quit): **1234567**

1234567 % 256 is 135

Enter next number for first operand (<= 0 to quit): **0**

Done

9. Write a program that requests the user to enter a Fahrenheit temperature. The program should read the temperature as a type double number and pass it as an argument to a user-supplied function called Temperatures() . This function should calculate the Celsius equivalent and the Kelvin equivalent and display all three temperatures with a precision of two places to the right of the decimal. It should identify each value with the temperature scale it represents. Here is the formula for converting Fahrenheit to Celsius:

Celsius = 5.0 / 9.0 * (Fahrenheit - 32.0)

The Kelvin scale, commonly used in science, is a scale in which 0 represents absolute zero, the lower limit to possible temperatures. Here is the formula for converting Celsius to Kelvin:

Kelvin = Celsius + 273.16

The Temperatures() function should use const to create symbolic representations of the three constants that appear in the conversions. The main() function should use a loop to allow the user to enter temperatures repeatedly, stopping when a q or other nonnumeric value is entered. Use the fact that scanf() returns the number of items read, so it will return 1 if it reads a number, but it won't return 1 if the user enters q. The == operator tests for equality, so you can use it to compare the return value of scanf() with 1 .