Computer Science
Chapter 6 Program Set (Functions)

Program 1 - *Rounding Numbers*

The floor function can be used to round a number to a specific decimal place.
The statement

*floor(x * 10 + .5) / 10;*

rounds x to the tenths position, and the statement

*floor(x * 100 + .5) / 100;*

rounds x to the hundredths position.

Write a program that defines four functions to round a number x in various ways:

roundToInteger(number)
roundToTenths(number)
roundToHundredths(number)
roundToThousandths(number)

Program 2 – *Random Numbers in a Sequence*

Write a program that defines three functions comprised of **exactly one single statement each** that returns a number at random from each of the following sets:

functionA() prints a random number from 2, 4, 6, 8, 10
functionB() prints a random number from 3, 5, 7, 9, 11
functionC() prints a random number from 6, 10, 14, 18, 22

Program 3 – *Multiples*

Write a Boolean function *multiple* that determines for a pair of integers whether the second is a multiple of the first. The function should take two integer arguments and return true if the second is a multiple of the first, false otherwise.

Program 4– *Number of Seconds*

Write an integer function that takes the time as three integer arguments (hours, minutes and seconds) and returns the number of seconds since midnight.

Use this function to calculate the amount of time in seconds between two times, both of which are within one 24-hour cycle of the clock.

Program 5 – *Temperature*

Implement the following integer functions:

Function celsius returns the Celsius equivalent of a Fahrenheit temperature.
Function fahrenheit returns the Fahrenheit equivalent of a Celsius temperature.

Use these functions to write a program that prints charts showing the Fahrenheit equivalents of all Celsius temperatures from 0 to 100 degrees, and the Celsius equivalents of all Fahrenheit temperatures from 32 to 212 degrees. Print the outputs in a table.

Program 6 – *Perfect Numbers*

An integer is said to be a *perfect number* if the sum of its divisors, including 1 (but not the number itself), is equal to the number. For example, 6 is a perfect number, because 6 = 1 + 2 + 3.

Write a Boolean function *isPerfect* that determines whether a number is a perfect number.

Use this function in a program that determines and prints all the perfect numbers between 1 and 10000.

Program 7 – GCD

The *greatest common divisor (GCD)* of two integers is the largest integer that evenly divides each of the numbers. Write a function *gcd* that returns the greatest common divisor of two integers

Program 8 – Letter or Nah

Create a Boolean function isLetter which takes a character as a parameter and returns true or false if the character is a letter or not – it can be capital or lower case. Hint – use ASCII!

Program 9 – Area of a Polygon

Create a double function areaPolygon which takes two integers as a parameter – the number of sides of the polygon and the length of each side. The function will find the area of a regular polygon with these specifications. (Yes, there are formulas for this.)

Program 10 – Has Twin Prime

Create a Boolean function hasTwinPrime which takes an integer as a parameter and returns true or false if the character has a twin prime or not – either higher or lower. (Twin primes are two prime numbers which are separated by exactly two.)

For example, hasTwinPrime(43) would be true, as 41 and 43 are twin primes.

Program 11 – Count 5 (MUST BE RECURSIVE)

Given a positive integer, return the count of the number of fives as a digit in that number. For example, 515 would return 2. The number will be in the range of integers, and the function must be recursive for you to receive credit.

Program 12 – Triangle (MUST BE RECURSIVE)

Given a triangle made of blocks where the topmost row has 1 block, the next row down has 2 blocks, the next row has 3 blocks, and so on. Enter the number of rows of your triangle, and compute recursively the total number of blocks in such a triangle.