Practice Questions On Functions

1. Write a function that inputs a number and prints the multiplication table of that number

2. Write a program to print twin primes less than 1000. If two consecutive odd numbers are both prime then they are known as twin primes

3. Write a program to find out the prime factors of a number. Example: prime factors of 56 - 2, 2, 2, 7

4. Write a program to implement these formulae of permutations and combinations. Number of permutations of n objects taken r at a time: p(n, r) = n! / (n-r)!. Number of combinations of n objects taken r at a time is: c(n, r) = n! / (r!*(n-r)!) = p(n,r) / r!

5. Write a function that converts a decimal number to binary number

6. Write a function cubesum() that accepts an integer and returns the sum of the cubes of individual digits of that number. Use this function to make functions PrintArmstrong() and isArmstrong() to print Armstrong numbers and to find whether is an Armstrong number.

7. Write a function prodDigits() that inputs a number and returns the product of digits of that number.

8. If all digits of a number n are multiplied by each other repeating with the product, the one digit number obtained at last is called the multiplicative digital root of n. The number of times digits need to be multiplied to reach one digit is called the multiplicative persistance of n.
   Example: 86 -> 48 -> 32 -> 6 (MDR 6, MPersistence 3)
   341 -> 12->2 (MDR 2, MPersistence 2)
   Using the function prodDigits() of previous exercise write functions MDR() and MPersistence() that input a number and return its multiplicative digital root and multiplicative persistence respectively

6. Write a function cubesum() that accepts an integer and returns the sum of the cubes of individual digits of that number. Use this function to make functions PrintArmstrong() and isArmstrong() to print Armstrong numbers and to find whether is an Armstrong number.

7. Write a function prodDigits() that inputs a number and returns the product of digits of that number.

8. If all digits of a number n are multiplied by each other repeating with the product, the one digit number obtained at last is called the multiplicative digital root of n. The number of times digits need to be multiplied to reach one digit is called the multiplicative persistance of n.
   Example: 86 -> 48 -> 32 -> 6 (MDR 6, MPersistence 3)
   341 -> 12->2 (MDR 2, MPersistence 2)
   Using the function prodDigits() of previous exercise write functions MDR() and MPersistence() that input a number and return its multiplicative digital root and multiplicative persistence respectively

9. Write a function sumPdivisors() that finds the sum of proper divisors of a number. Proper divisors of a number are those numbers by which the number is divisible, except the number itself. For example proper divisors of 36 are 1, 2, 3, 4, 6, 9, 18

10. A number is called perfect if the sum of proper divisors of that number is equal to the number. For example 28 is perfect number, since 1+2+4+7+14=28. Write a program to print all the perfect numbers in a given range

11. Two different numbers are called amicable numbers if the sum of the proper divisors of each is equal to the other number. For example 220 and 284 are amicable numbers.

    Sum of proper divisors of 220 = 1+2+4+5+10+11+20+22+44+55+110 = 284
    Sum of proper divisors of 284 = 1+2+4+71+142 = 220
    Write a function to print pairs of amicable numbers in a range

12. Write a program which can filter odd numbers in a list by using filter function

13. Write a program which can map() to make a list whose elements are cube of elements in a given list

14. Write a program which can map() and filter() to make a list whose elements are cube of even number in a given list

15.