Practice Questions

1. Program finds the sum of the first 100 positive integers and also their average.

2. Find the sum of the 10 positive integers entered by the user if the user enters negative number the loop should be terminated and the result should be displayed.

3. Find the sum of the 10 positive integers entered by the user if the user enters negative number that iteration should be skipped without incrementing the loop control value.

4. Write a program that asks the user for char values and sum the ASCII values of all the entered characters the loop should be stopped when the user enters the **Space character**.

5. Find the sum of the first 10 numbers that are divisible by 3 and 9.

6. Write a program that asks for an integer value of any length and find the number of zeros in the integer.

7. In question 5 if the sum increase than 200 break the loop.

8. Write a program that asks the user to type 10 integers and write the number of occurrence of the biggest value.

9. Write a program that asks the user to type 10 integers and writes the smallest value.

10. Write a program that calculate power of a number using loops.

11. Write a program that prompts the user to input a positive integer. It should then output a message indicating whether the number is a prime number. (Note: An even number is prime if it is 2. An odd integer is prime if it is not divisible by any odd integer less than or equal to the square root of the number.)

12. Write a program that reads a set of integers and then finds and prints the sum of the even and odd integers.

13. Write a program that as for an integer value from the user and then prints the length of the value using loop.

e.g :

```
Enter any integer Value : 78654321
The length of the Entered Value : 8
```

14. Write a program that calculates factorials. The factorial of a positive integer number is mathematically written as and is the product of all numbers from 1 to n; Mathematically, if the number is 7 the factorial will be 7*6*5*4*3*2*1=5040

15.
```
*
**
***
****
*****
```

16.
```
    *
   **
  ***
 ****
*****
```

17. Write a program that asks the user to type a positive integer. When the user types a negative value the program writes ERROR and asks for another value. When the user types 0 that means that the last value has been typed and the program must write the average of the positive integers. If the number of typed values is zero the program writes 'NO AVERAGE'.

18.
```
    *
   **
  ***
 ****
*****
```

19.
```
    *
    **
   ***
  ****
 *****
  ****
   ***
    **
    *
```

20.
```
*****
 ****
  ***
   **
    *
   **
  ***
 ****
```

Write a function named "sum_from_to" that takes two integer arguments, call them "first" and "last", and returns as its value the sum of all the integers between first and last inclusive. Thus, for example, cout << sum_from_to(4,7) << endl;//will print 22 because 4+5+6+7 = 22
cout << sum_from_to(-3,1) << endl;//will print -5 'cause (-3)+(-2)+(-1)+0+1 = -5 cout << sum_from_to(7,4) << endl;// will print 22 because 7+6+5+4 = 22
cout << sum_from_to(9,9) << endl;// will print 9

Write a function named "enough" that takes one integer argument, call it "goal" and returns as its value the smallest *positive* integer n for which 1+2+3+. . . +n is at least equal to goal . Thus, for example,
cout << enough(9) << endl; //will print 4 because 1+2+3+4 9 but 1+2+3<9
cout << enough(21) << endl;//will print 6 'cause 1+2+ . . .+6 21 but 1+2+ . . . 5<21

Write a function named "g_c_d" that takes two *positive* integer arguments and returns as its value the greatest common divisor of those two integers. If the function is passed an argument that is not positive (i.e. Greater than zero), then the function should return the value 0 as a sentinel value to indicate that an error occurred. Thus, for example,
cout << g_c_d(40,50) << endl; // will print 10 cout << g_c_d(256,625) << endl; // will print 1
cout << g_c_d(42,6) << endl; // will print 6

A positive integer n is said to be ***prime*** (or, "a prime") if and only if n is *greater than* 1 and is divisible only by 1 and n. For example, the integers 17 and 29 are prime, but 1 and 38 are not prime. Write a function named "is_prime" that takes a *positive* integer argument and returns as its value the integer 1 if the argument is prime and returns the integer 0 otherwise. Thus, for example,
cout << is_prime(19) << endl; // will print 1 cout << is_prime(1) << endl; // will print 0 cout << is_prime(51) << endl; // will print 0 cout << is_prime(-13) << endl; // will print 0

Write a function named "digit_name" that takes an integer argument in the range from 1 to 9, inclusive, and prints the English name for that integer on the computer screen. No newline character should be sent to the screen following the digit name. The function should not return a value. The cursor should remain on the same line as the name that has been printed. If the argument is not in the required range, then the function should print "digit_error" without the quotation marks but followed by the newline character. Thus, for example,
The statement digit_name (7); should print seven on the screen;
The statement digit_name (0); should print digit_error on the screen and place the cursor at the beginning of the next line.

Write a function named "reduce" that takes two positive integer arguments, call them "num" and "denom", treats them as the numerator and denominator of a fraction, and reduces the fraction. That is to say, each of the two arguments will be modified by dividing it by the greatest common divisor of the two integers. The function should return the value 0 (to indicate failure to reduce) if either of the two arguments is zero or negative, and should return the value 1 otherwise. Thus, for example, if m and n have been declared to be integer variables in a program, then
m = 25;
n = 15;
if (reduce(m,n))
cout << m << '/' << n << endl; else
cout << "fraction error" << endl; will produce the following output: 5/3
Here is another example. m = 25;
n = 0;
if (reduce(m,n))
cout << m << '/' << n << endl; else
cout << "fraction error" << endl; will produce the following output: fraction error
The function reduce is allowed to make calls to other functions that you have written.

Write a function named "swap_floats" that takes two floating point arguments and interchanges the values that are stored in those arguments. The function should return no value. To take an example, if the following code fragment is executed
float x = 5.8, y = 0.9; swap_floats (x, y);
cout << x << " " << y << endl; then the output will be
0.9 5.8

Write a function named "sort3" that takes three floating point arguments, call them "x" , "y" , and "z", and modifies their values, if necessary, in such a way as to make true the following inequalities: x <y < z . The function should return no value. To take an example, if the following code fragment is executed

float a = 3.2, b = 5.8, c = 0.9; sort3 (a, b, c);
cout << a << " " << b << " " << c << endl; then the output will be 0.9 3.2 5.8
The function sort3 is allowed to make calls to other functions that you have written.