**In each exercise make your source code and output readable.**

**Exercise 1.** Write a program that outputs the sentence "Learn to walk before you run."
VERSION 1. Repeat the sentence n times, where n is given by the user.
VERSION 2. Repeat the sentence as many times as the user wants, ask the user after each run of the program whether to display the sentence once again.

**Exercise 2.** A large company allocates its employees leave based on the number of hours worked in a week. Each employee gets two hours of leave for each week worked plus 10% of hours worked. Develop a C++ program that uses a loop to input each employee's hours worked for last week and calculates and displays the number of hours of leave accrued by the employee. Process each employee's figures at a time. Repeat calculations until the value -1 for hours worked in a week is given. Sample run:

```
Enter number of hours worked (-1 to end): 10.5
Accrued leave: 3.05 hours

Enter number of hours worked (-1 to end): 45
Accrued leave: 6.50 hours

Enter number of hours worked (-1 to end): 30
Accrued leave: 5.00 hours

Enter number of hours worked (-1 to end): -1
```

MODIFICATION. Round the number of hours of leave accrued by the employee to the nearest integer. Use round function from cmath library.

**Exercise 3.** Write a program that predicts the size of a population of organisms. The program should ask for the starting number of organisms, their average daily population increase (as a percentage), and the number of days they will multiply. For example, a population might begin with two organisms, have an average daily increase of 50 percent, and will be allowed to multiply for seven days. The program should use a loop to display the size of the population for each day. So taking 2 organisms, 50% as an average daily increase and 7 days, program should output: day #1: 2 day #2: 3 day #3: 4.5 day #4: 6.75 day #5: 10.125 day #6: 15.1875 day #7: 22.78125
VERSION 1. Repeat calculations for n organisms, where n is given by the user.
VERSION 2. Repeat calculations as many times as the user wants.
MODIFICATION. Input Validation: Do not accept a number less than 2 for the starting size of the population. Do not accept a negative number for average daily population increase. Do not accept a number less than 1 for the number of days they will multiply.

**Exercise 4.** Write a program that asks the user to enter the amount of their monthly budget. A loop should then prompt the user to enter each of the users expenses for the month and keep a running total. When the loop finishes, the program should display the amount that the user is over or under budget.
Sample run #1:

```
Enter amount budgeted for the month: 1250.00
Enter an amount spent (0 to end): 145.87
Enter an amount spent (0 to end): 15
Enter an amount spent (0 to end): 198
Enter an amount spent (0 to end): 345.99
Enter an amount spent (0 to end): 0
Budgeted: 1250.00
Spent: 704.86
You are 545.14 under budget. WELL DONE!
```

Sample run #2:

```
Enter amount budgeted for the month: 759
Enter an amount spent (0 to end): 127.43
Enter an amount spent (0 to end): 150
Enter an amount spent (0 to end): 600
Enter an amount spent (0 to end): 0
Budgeted: 759.00
Spent: 877.43
You are 118.43 over budget. PLAN BETTER NEXT TIME!
```

**Exercise 5.** Write a program that that reads real numbers from the input until the value 0 is given. If the real number is positive, then program displays square root of the number otherwise the program displays square of the number. In addition, the program counts how many real numbers have been given and how many of them are positive and how many are negative. In a program use pow and sqrt functions from cmath library.

Interaction with the program might look like this:

```
Enter real numbers. Zero ends reading.
1. Give a number: 3
Square root: 1.73205
2. Give a number: -4
Square: 16
3. Give a number: -2
Square: 4
4. Give a number: 5
Square root: 2.23607
5. Give a number: 0
READING ENDS
5 numbers are given (2 positive and 2 negative)
```

MODIFICATION 1. Additionally, program calculates and displays two sums: the sum of positive numbers and the sum of negative numbers.

MODIFICATION 2. Additionally, program calculates and displays two arithmetic averages: the arithmetic average of positive numbers and the arithmetic average of negative numbers.

**Exercise 6. (A bit complex)** Write a program that reads the integers from the input until the value 0 is given, and prints these subsequent consecutive pairs whose product is greater than an integer value G given by the user. Sample run for G=0

```
1. Give a number: 3
2. Give a number: -4
3. Give a number: -2
Pair #1: -4 -2
4. Give a number: -5
Pair #2: -2 -5
5. Give a number: 4
6. Give a number: 2
Pair #3: 4 2
7. Give a number: -5
8. Give a number: 0
READING ENDS
```

**Exercise 7. (A bit complex)** Write a program that reads the integers from the input until the value 0 is given, and prints these subsequent consecutive triples ordered in increasing order.

```
1. Give a number: 3
2. Give a number: -3
3. Give a number: -2
4. Give a number: -1
Triple #1: -3 -2 -1
5. Give a number: 4
Triple #2: -2 -1 4
6. Give a number: 2
7. Give a number: -5
8. Give a number: 0
READING ENDS
```