3. Loops (while/for/do while). Nested loops.

3.01. Fizz buzz is a group word game for children to teach them about division. Players take turns to count incrementally, replacing any number divisible by three with the word "fizz", and any number divisible by five with the word "buzz". Numbers divisible by 15, which is both become fizz buzz. For example, a typical round of fizz buzz would start as follows:

1, 2, Fizz, 4, Buzz, Fizz, 7, 8, Fizz, Buzz, 11, Fizz, 13, 14, Fizz Buzz, 16, 17, Fizz, 19, Buzz, Fizz, 22, 23, Fizz, Buzz, 26, Fizz, 28, 29, Fizz Buzz, 31, 32, Fizz, 34, Buzz, Fizz, ...

Writing a program to output the first 100 FizzBuzz numbers.

3.02. Write a program which calculates the arithmetic mean of the digits of a given integer long long number, formatted to the second digit after the decimal point.

3.03. Write a function with signature int digitPos(long long num, int k), which returns the k-th digit of the number num or -1 if such position does not exist. It starts counting from left to right and from 1. For example: digitPos(1234, 3) returns 3.

3.04. Write a function with signature double sqrt_x(double x, double eps), which calculates a square root of x with an epsilon (eps) precision.

3.05. Write a program which asks the user for a long long number and then prints:

- the same number in binary;
- the same number in oct;
- the same number in hex.

3.06. Write a program which asks the user for a number n and then prints whether n is a prime number or not.

3.07 Write a program wchich asks the user for a number n and then prints all prime numbers smaller than n.

3.08. Write a program which asks the user for a number n and then prints all prime factors of n.

Example input	Expected output
225	3.3.5.5
31668	2.2.3.7.13.29

3.09. Write a program which checks whether a number can be expressed as a sum of two prime numbers.

3.10. Wtire a program which asks the user for a number n and then prints a square with numbers with as many rows as n. Align all numbers in the square!

• For n=3, the output must be;

```
1 2 3
4 5 6
7 8 9
```

• For n=4, the output must be (note how all columns are aligned)

```
1  2  3  4
5  6  7  8
9  10 11 12
13 14 15 16
```

3.11. Wtire a program which asks the user for a number n and then prints a triangle with numbers with as many rows as n. This right-angled triangular array of natural numbers is named after Robert Floyd. Align all numbers in the triangle!

• For n=4, the output must be;

```
1
2 3
4 5 6
7 8 9 10
```

• For n=7, the output must be (note how all columns are aligned)

```
1
2  3
4  5  6
7  8  9 10
11 12 13 14 15
16 17 18 19 20 21
22 23 24 25 26 27 28
```

3.12. Consider the sequence: 1;2+3;4+5+6;7+8+9+10;... . In other words, the n-th term of the sequence is the "sum of the next n natural numbers" - Felice Russo. Write a program which prints the first n terms of the sequence.

3.13. Write a program which checks to which row and column in the Floyd's triangle the number n belongs.

3.14. Wtire a program which asks the user for an odd number n and then prints a Christmas tree as wide as n. Print and error if n is not odd!

• For n=5, the output must be:

```
  *
 ***
*****
  *
```

• For n=7, the output must be:

```
   *
  ***
 *****
*******
   *
```

3.15. Write a program which asks the user for an odd number n and then prints a heart as wide as 2n+1. Print and error if n is not odd!

• For n=3, the output must be:

```
 *   *
*** ***
 *****
  ***
   *
```

• For n=5, the output must be:

```
  *     *
 ***   ***
***** *****
 *********
  *******
   *****
    ***
     *
```

3.16. Ask a friend to take a piece of paper and ***write*** down a number from 1 to 1000, without showing it to you. Write a program that guesses the written number only by asking questions like **“Is your number bigger than {arbitrary number of your choice}?”**, etc. if your question contains the correct answer, then your friend needs to answer with **“that's the number!”**, otherwise with **“yes”** or **“no”.** Assuming your friend is not lying, how many questions will your program ask in the worst case?

3.17. Write a function with signature *double calc(double a, double b, double eps, long end)*, which calculates

$x_k = \frac{1}{5}\left(x_{k-1} + \frac{x_{k-1}}{x_{k-2}}\right), k = 3,4,...; x_1 = a > 0, x_2 = b$. The calculations end when $|x_k - x_{k-1}| < \epsilon$. If k becomes greater than n - the function must return a code for error.

3.18. Write a program which calculates the sum $e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \ldots$ and find the Napier's constant. The approximation continues until the absolute value of the last collectible becomes less than ϵ (where x and ϵ are given real numbers).

3.19. Find the smallest number expressible as the sum of two cubes in two different ways and print the expressions. ([Hardy–Ramanujan number](https://en.wikipedia.org/wiki/1729_(number)))

3.20. Write a program that reads an integer number N and prints how many trailing zeros are present at the end of the number N! (N factorial). Make sure your program works for N up to 50 000. Hint: you don't need to calculate the actual factorial to count the number of trailing zeroes.

Example input	Expected output	Explanation
1	0	1!=1, no trailing zeros
5	1	5!=120, 1 trailing zero
20	4	20!=2432902008176640000, 4 trailing zeros