

Exercise 1: Multiplication Tables

1. Print every multiplication table from 1*1 to 10*10.
2. Print only the odd result (i.e.: 3*7 = 21). Tip: use the modulo (%) operator.
3. Add the feature to give a parameter, the display the multiplication table for this number **N**.
You can use this function to ask a parameter to the user.

```
private static int AskUserForParameter()  
{  
    Console.WriteLine("Please write a number and press enter :");  
    int.TryParse(Console.ReadLine(), out var result);  
    return result;  
}
```

Exercise 2: More math

Now for a little bit of algorithmic... Using mathematical libraries given by the system is forbidden.

1. Write a function **Prime()** that prints all prime number between 1 and 1000.
2. Write a Fibonacci function **F** with **F(0) = 1**, **F(1) = 1** and **F(N) = F(N-1) + F(N-2)** for **N >= 2**.
The number **N** is determined by asking a value to the user through the console.
3. Write a factorial function. Reminder:
 - a. $4! = 4 \cdot 3 \cdot 2 \cdot 1 = 24$
 - b. $6! = 6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 = 720$
 - c. What would happen if you tried to calculate 420.000!?
 - d. Theoretical: what is recursive function?

Exercise 3: Try/Catch

Use your Googling skills to read information about the **try/catch** block and implement one that divides 10 by the result of this function for every value between -3 and 3 (using a for loop).

$$\Rightarrow F(x) = 10/(x^2-4) \text{ with } -3 \leq x \leq 3$$

```
private static int PowerFunction(int x)  
{  
    return (int) (Math.Pow(x, 2) - 4);  
}
```

This should result in a DividedByZero Exception on $x = -2$ and $x = 2$.

Yet, at no point must your program crash or stop. Write (or not) the exception and continue the program. This kind of block will be crucial for your exam: while reading an API, if something goes south, you'll have to make sure that your program stays up and running to try again seconds later.

Exercise 4: Square

1. Fun times... Print a **N** by **M** rectangle, having the following properties:

- **N** and **M** are ≥ 1 and ≤ 1000
- Corners are "0"
- Horizontal lines are "-"
- Vertical lines are "|"

Know that a (1,1) is just "0".

What does a (1,5); (4,1); (5,7) or (3,3) rectangle look like?

Please ask the user for both **N** and **M** at once.

2. Inside the rectangle, add a "*" and two spaces so that stars are in a diagonal pattern.

Expected output for (9,6):

```
0-----0
|*   *   *|
|*   *   *|
| *   *   *|
|*   *   *|
0-----0
```

Exercise 5: Christmas Tree

Print a Christmas Tree! It should look like this:

```
      *
     ***
    *****
   ********
  **********
 ***
**
| |
```

Regular tree

```
      *
     i**
    ***o***
   *i**o**
  i**i**i**
 ***o**i**i**
   | |
```

Decorated tree

Tree size must be asked to the user. In this example, this tree is of size 6. Size must be between 3 and 20. Don't forget to print the trunk with "|".

Then you need to decorate it! Following the pattern in the example, change some of the star by a 'i' and a 'o' here and there! Decoration must be an option for the user. Don't have too many 'o', it must stay a lesser number than 'i'.

Good luck!