

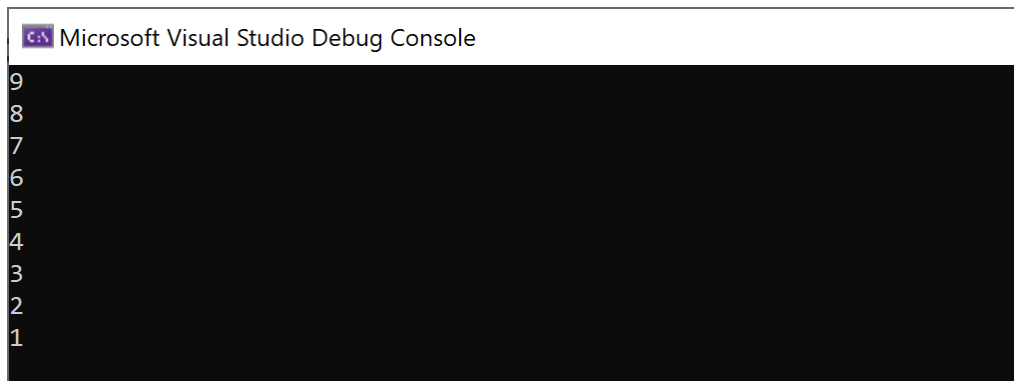
C# general tasks

Set 2

Note: for all these exercises, ignore input validation unless otherwise directed. Assume the user enters a value in the format that the program expects. For example, if the program expects the user to enter a number, don't worry about validating if the input is a number or not. When testing your program, simply enter a number.

Task 2.1

Write a program that, *by using a for loop*, adds the numbers 1 to 9 to an int array of size 9. (Not size 10 or anything else.) It should then output the array backwards, one element per line, by using another for-loop. Here is an example of possible output:



```
Microsoft Visual Studio Debug Console
9
8
7
6
5
4
3
2
1
```

Extra: It's also possible to reverse the contents of a full array, by calling a method that does exactly that. If you want to, see if you figure out what the method is called and how to use it. If so, show this as an extra solution for task 2.1.

Task 2.2

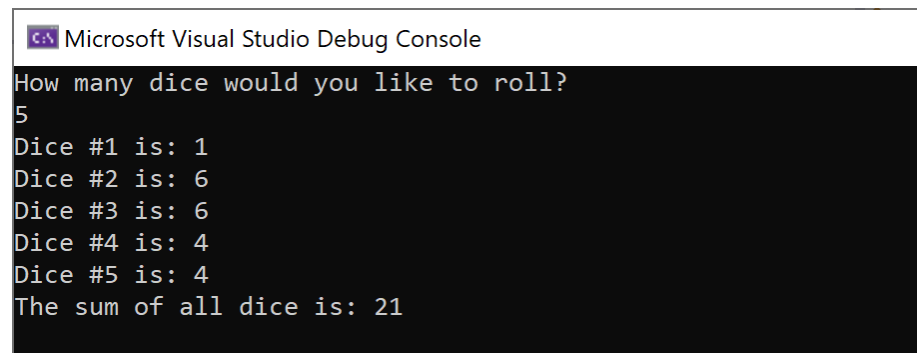
Random numbers in C# are handled by creating an instance of the Random class:

```
var random = new Random();
```

The following code will set number to a random value of 0, 1, 2 or 3:

```
int number = random.Next(0, 4); // 0 or more, but less than 4.
```

Simulate the throw of a dice. (Should give the value 1, 2, 3, 4, 5, or 6.) Write a program that asks the user how many dice she or he wants to throw. Then displays the result of each dice, as well as the sum of the value on all dice added together. Example:

A screenshot of the Microsoft Visual Studio Debug Console. The title bar at the top says "Microsoft Visual Studio Debug Console". The console output shows a prompt "How many dice would you like to roll?" followed by the user input "5". Then, it displays the results for five dice: "Dice #1 is: 1", "Dice #2 is: 6", "Dice #3 is: 6", "Dice #4 is: 4", and "Dice #5 is: 4". Finally, it shows the total: "The sum of all dice is: 21".

```
Microsoft Visual Studio Debug Console
How many dice would you like to roll?
5
Dice #1 is: 1
Dice #2 is: 6
Dice #3 is: 6
Dice #4 is: 4
Dice #5 is: 4
The sum of all dice is: 21
```

Extra: The version shown above is made by summing the total for each dice thrown. However, it's also possible to put the result of each dice roll in a list or array, and sum them in the end, with a separate method for that. If you want to, see if you figure out how that is done and hand it in as an extra solution for task 2.2.

Task 2.3

Create a function that writes out N numbers of the Fibonacci sequence, where N is 2 or more. The next number in the Fibonacci sequence is the sum of the two numbers prior. To get started, we have the two first values defined as 0 and 1. The sequence looks like this:
0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ...

Here's a suggestion for a possible approach to do this:

1. Set two variables to the first two values (0 and 1), as well as add them to a string that will become the full sequence when we are done.
2. Create a loop that iterates from 2 (since we already have the first 2 numbers) to N.
3. Inside the loop:
 - a. Sum the 2 previous values in a third variable and write it to our string.
 - b. update the 2 previous values to be the new sum and the value before it.
 - c. Keep calculating new sums and updating previous values, until we have N numbers in the sequence.
4. Outside the loop again, write the string with the full sequence to the console.

Example:

```
Microsoft Visual Studio Debug Console
How many numbers in the Fibonacci sequence would you like to calculate?
18
The sequence is:
0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597
```

Extra: As for the other tasks this week, also this one can be done with an array or a list. The approach would be something like:

1. Create a list or array, add 2 elements containing 0 and 1 to it.
2. Through a loop, keep adding as many Fibonacci-elements as needed to make the sequence the user asked for.
3. To print the contents of the full list/array in an easy and elegant manner, look into how the `string.Join()` method works.