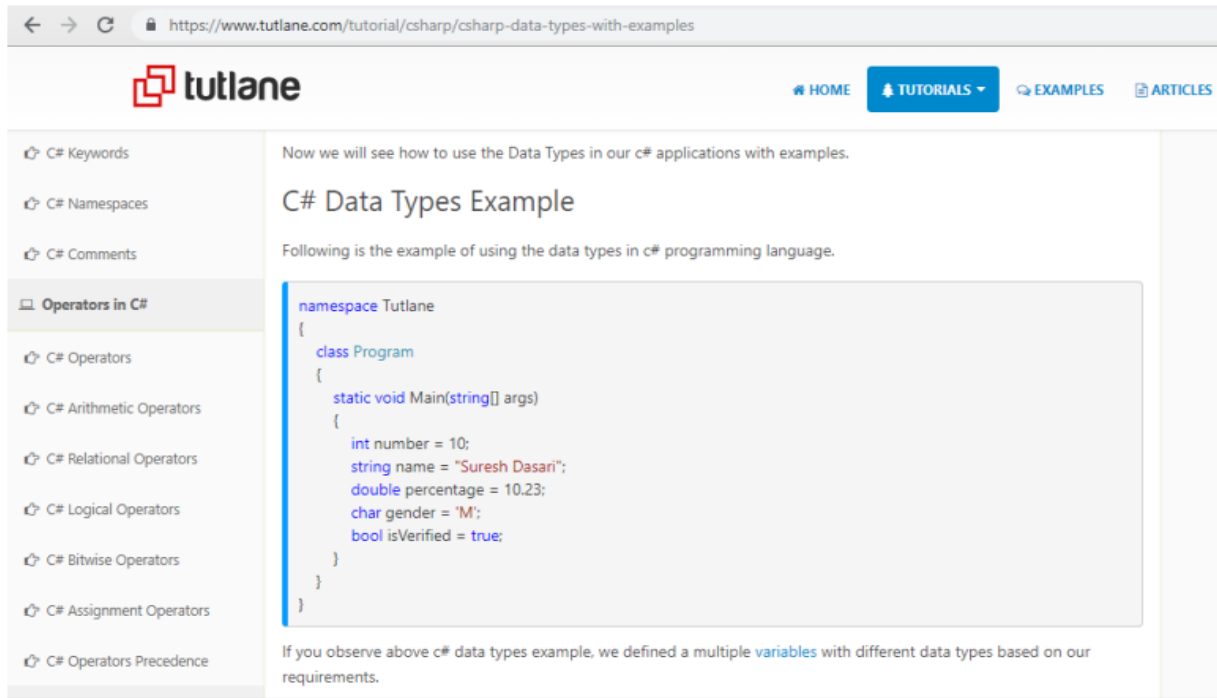


Recap

- Continue to write the following programs with Comments incl. Declarations, Input, Output & Processing
- Variables Types



Now we will see how to use the Data Types in our c# applications with examples.

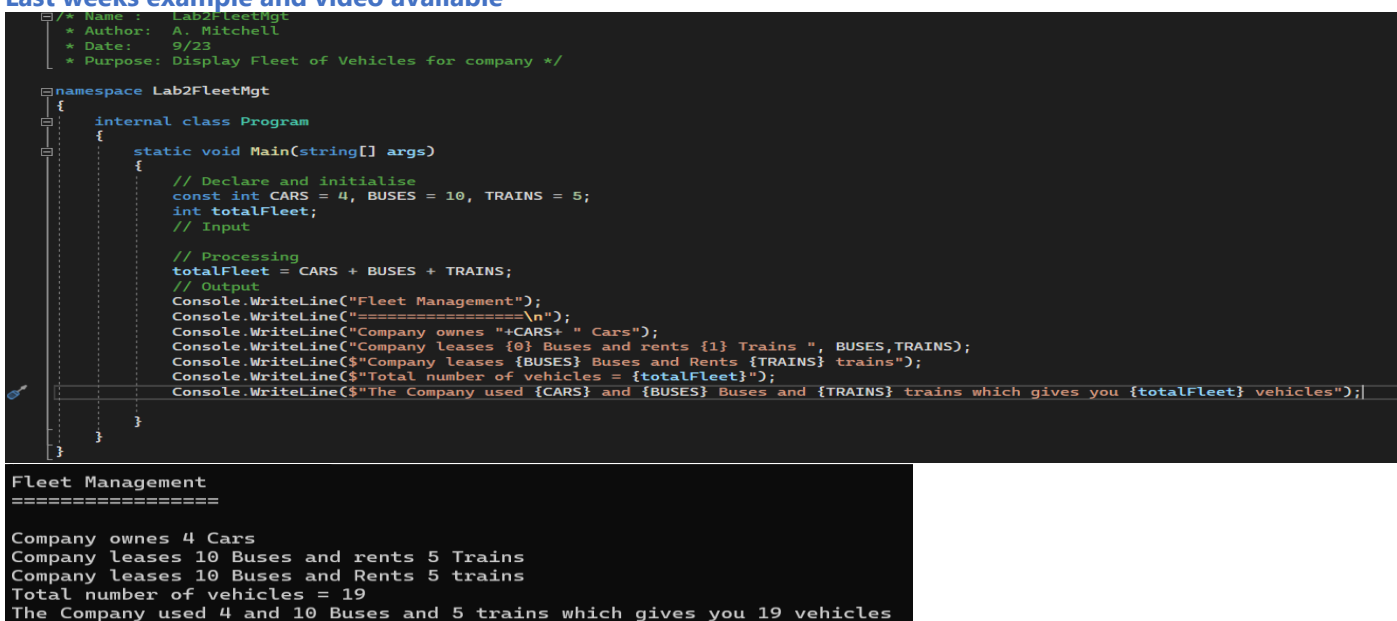
C# Data Types Example

Following is the example of using the data types in c# programming language.

```
namespace Tutlane
{
    class Program
    {
        static void Main(string[] args)
        {
            int number = 10;
            string name = "Suresh Dasari";
            double percentage = 10.23;
            char gender = 'M';
            bool isVerified = true;
        }
    }
}
```

If you observe above c# data types example, we defined a multiple variables with different data types based on our requirements.

- **Constant fields (CONST)** In C#, a **const** keyword is used to declare **constant** fields and **constant** local. The value of the **constant** field is the same throughout the program or in other words, once the **constant** field is assigned the value of this field is not be changed. Constants are immutable (cannot be modified) values which are known at compile time and do not change for the life of the program.
- **Output** `Console.WriteLine("Hi " + nameInput + ", you are " + ageInteger + " years old.");`
- **Example using Placeholders for output**
`Console.WriteLine("Hi {0}, you are {1} years old.", nameInput, ageInteger);`
*** {0} Placeholders always start at 0* <https://www.youtube.com/watch?v=fdkckML-yG0>
`Console.WriteLine($"Hi {nameInput}, you are {ageInteger} years old,);`
- **Last weeks example and video available**



```
/* Name : Lab2FleetMgt
 * Author: A. Mitchell
 * Date: 9/23
 * Purpose: Display Fleet of Vehicles for company */

namespace Lab2FleetMgt
{
    internal class Program
    {
        static void Main(string[] args)
        {
            // Declare and initialise
            const int CARS = 4, BUSES = 10, TRAINS = 5;
            int totalFleet;
            // Input

            // Processing
            totalFleet = CARS + BUSES + TRAINS;
            // Output
            Console.WriteLine("Fleet Management");
            Console.WriteLine("=====\n");
            Console.WriteLine("Company owns "+CARS+ " Cars");
            Console.WriteLine("Company leases {0} Buses and rents {1} Trains ", BUSES, TRAINS);
            Console.WriteLine($"Company Leases {BUSES} Buses and Rents {TRAINS} trains");
            Console.WriteLine($"Total number of vehicles = {totalFleet}");
            Console.WriteLine($"The Company used {CARS} and {BUSES} Buses and {TRAINS} trains which gives you {totalFleet} vehicles");
        }
    }
}
```

```
Fleet Management
=====

Company owns 4 Cars
Company leases 10 Buses and rents 5 Trains
Company leases 10 Buses and Rents 5 trains
Total number of vehicles = 19
The Company used 4 and 10 Buses and 5 trains which gives you 19 vehicles
```

Continue to write the following programs with Comments incl. Declarations constants where required, Input, Output with placeholders & Processing

Arithmetic – order of precedence

	Operator	Example
Add	+	a + b
Subtract	-	a - b
Multiply	*	a * b
Divide	/	a / b
Remainder	%	a % b

Operator Precedence with

Precedence rules for Arithmetic Operators

1. () parentheses
2. - (unary negation)
3. *, / , %
4. + -
5. If equal precedence, left to right

Adding Parentheses

You can change the order imposed by operator precedence and associativity by using parentheses. For example, `2 + 3 * 2` ordinarily evaluates to 8, because multiplicative operators take precedence over additive operators. However, if you write the expression as `(2 + 3) * 2`, the addition is evaluated before the multiplication, and the result is 10. The following examples illustrate the order of evaluation in parenthesized expressions. As in previous examples, the operands are evaluated before the operator is applied.

Statement	Order of evaluation
<code>a = (b + c) * d</code>	a, b, c, +, d, *, =
<code>a = b - (c + d)</code>	a, b, c, d, +, -, =
<code>a = (b + c) * (d - e)</code>	a, b, c, +, d, e, -, *, =

```
int myVariable = 5 + 7 * 3;
```

multiplication has higher precedence than addition, so $5+7 \times 3$ is equal to 26

```
int result;
result = 5 + 3 * 9; // result = 32
result = (5 + 3) * 9; // result = 72
```

Q1

Write a C# Sharp program to print the result of the specified operations. **Test data:** Check your answers with calculator

```
Console.WriteLine(2 + 3 * 5); // ans 17
Console.WriteLine(-1+4*6 + " is ans to 1+4*6 "); // ans 23
Console.WriteLine((-1+4)*6 + " is ans to (-1 + 4 )* 6 "); //ans 18
```

- $-1 + 4 * 6$
- $(-1 + 4) * 6$
- $((21 + 5) / 7)$
- $((21 + 5) \% 7)$
- $21 \% 10$
- $14 + -4 * 6 / 11$
- $2 + 15 / 6 * 1 - 7 \% 2$

Expected Output:

```
23 is ans to 1+4*6
18 is ans to (-1 + 4 )* 6
3 is ans to (21 + 5) / 7
5 is ans to (21 + 5) % 7
1 is ans to (21 % 10)
12 is ans to 14 + -4 * 6 / 11
3 is ans to 2 + 15 / 6 * 1 - 7 % 2
```

Write program to check others and check your answer on calculator

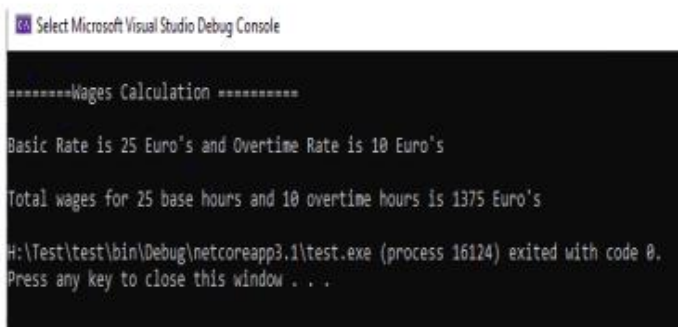
- $2+3*5$
- $9/4+10$
- $10/3$
- $21\%10$
- $(5-1)*3$
- $37/5$
- $64\%8$
- $5+2*4-23*4$
- $3*(2+5)/5$
- $28\%5-2$
- $19/2/2$
- $28/(2+4)$

Q2. Wages

Using variables to store all data, write a program to calculate hourly wages plus overtime. Base hourly rate is 25 euro, regular hours is 40, overtime rate is 37.5 euro and overtime hours is 10 . use constants and placeholders in solution

// Declare and Initialise variables

```
const int BASE_HR_RATE = 25, REG_HOURS=40, OVER_TIME_HOURS = 10;
const double OT_HOURLY_RATE = 37.5;
double hourlyWages = 0;
```



```
Select Microsoft Visual Studio Debug Console

*****Wages Calculation *****

Basic Rate is 25 Euro's and Overtime Rate is 10 Euro's

Total wages for 25 base hours and 10 overtime hours is 1375 Euro's

H:\Test\test\bin\Debug\netcoreapp3.1\test.exe (process 16124) exited with code 0.
Press any key to close this window . . .
```

Make a change to your program to display currency euro symbol €

```
static void Main(string[] args)
{
    Console.OutputEncoding = System.Text.Encoding.UTF8;
    //Declare
    double number = 1323.9, number2=.15;
    // Output
    Console.WriteLine($" \nDisplays output \n ");
    Console.WriteLine($" \t\tNumber\t\t{number:#,###.00} ");
    Console.WriteLine($" \t\tCurency\t\t{number:c} ");
    Console.WriteLine($" \t\tPercent\t\t{number2:p} ");
}
```

Displays output

```
Number  1,323.90
Curency €1,323.90
Percent 15.00%
```

Q3. Convert money

Using variables to store all data, write a program to convert 500 euro to sterling, exchange rate is 1 euro = 0.92 sterling **use constant's and placeholders in solution**

Microsoft Visual Studio Debug Console

```
=====Sterling Calculation =====  
  
Sterling Exchange rate is 0.92 and amount to be converted is 500  
  
500 Euro's converts to 460 Pounds Sterling
```

`const double CONVERT_RATE = .92, EUROS = 500.00;`

Modify program with exchange rate is 1 euro = 0.987

```
=====Sterling Calculation =====  
  
Sterling Exchange rate is 0.987 and amount to be converted is 500  
  
500 Euro's converts to 493.5 Pounds Sterling
```

Modify program to allow users to enter euro amount

Modify program to display € sign

Q4 Jeans Cost

Using variables to store all data, write a program to calculate the sale cost of a pair of jeans when the regular price is €125 and the sale is offering a 40% (.40) discount – **use constants and placeholders in solution**

// declare and initialise variable

`const double JEANS_PRICE = 125, DISCOUNT = .40;`

```
=====Jeans with % Discount Calculation =====  
  
Jeans cost 125 Euro's  
  
Discount of 40% on Jeans Valued 125 Euro's is 50 Euro's therefore the discounted Jeans cost 75 Euro's
```

Now change your program and change the discount at **25% (.25)**

// declare and initialise variable

`const double JEANS_PRICE = 125, DISCOUNT = .25;`

```
=====Jeans with % Discount Calculation =====  
  
Jeans cost 125 Euro's  
  
Discount of 25% on Jeans Valued 125 Euro's is 31.25 Euro's therefore the discounted Jeans cost 93.75 Euro's
```

Now change your program to display € and %

Q5 car fuel

Using variables to store all data, write a program to solve the following. Suppose certain car has a fuel tank with a capacity of 60 litres and an average fuel consumption of 14km/l. How many times must you fill the tank to travel 2000 km ?

use constants and placeholders in solution

How many km can you travel on 10 gallons of fuel using the same car?

1 gallons [UK] = 4.54609 liters

10 gallon [UK] to litre = 45.5 litre.