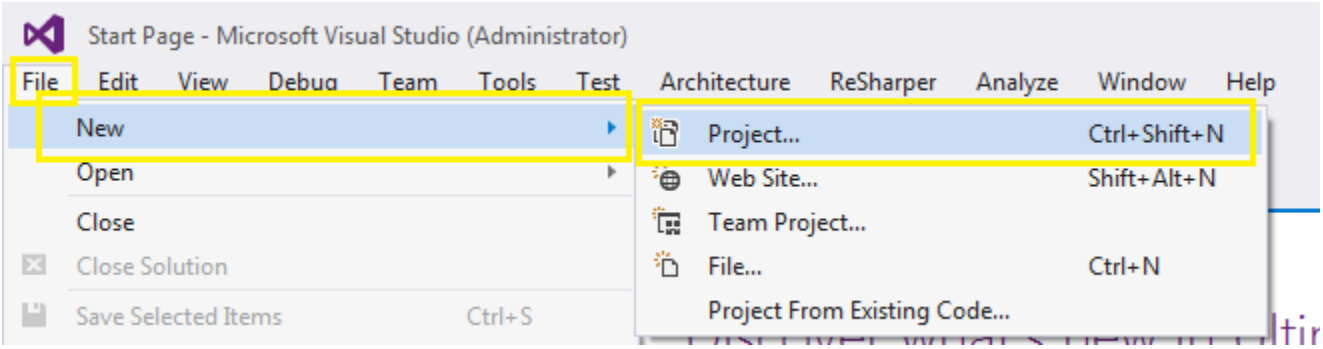
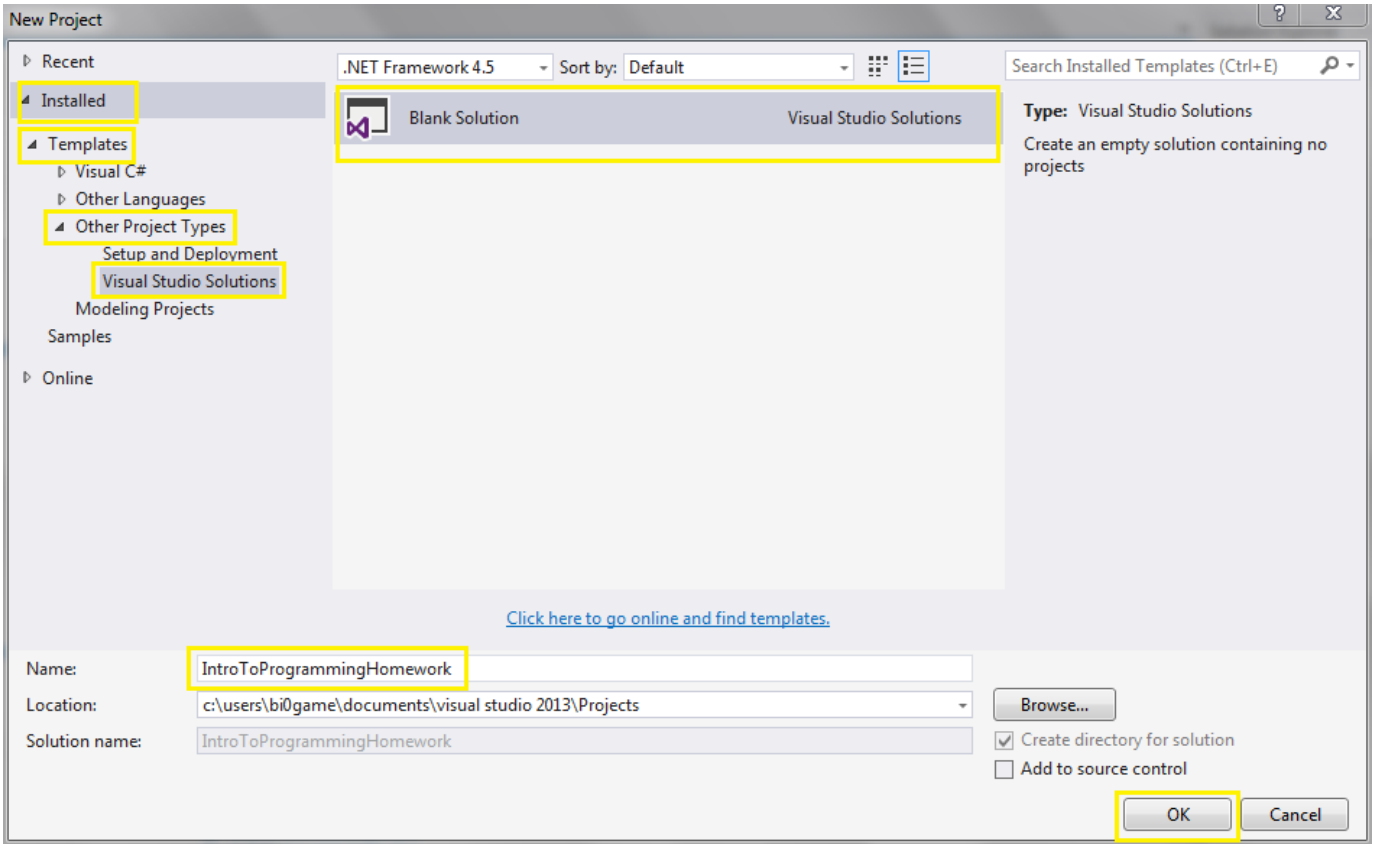
01. Introduction to Programming

Problem 1. Create A Blank Solution

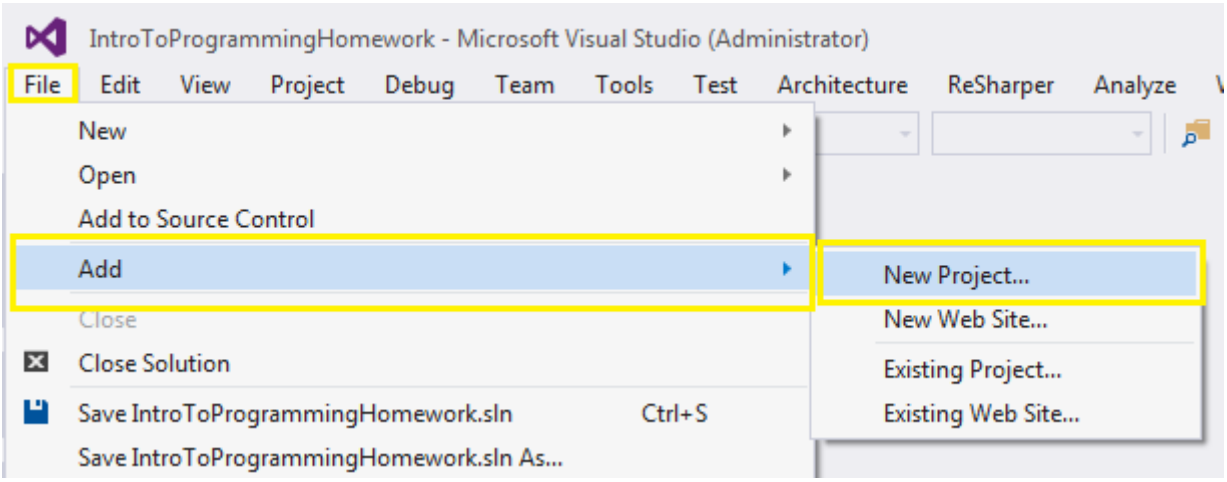
Open Visual Studio and select **File -> New -> Project**. Then from the menu to the left choose **Installed-> Templates -> Other Project Types -> Visual Studio Solutions -> Blank Solution**. Choose an appropriate name and click **OK**.

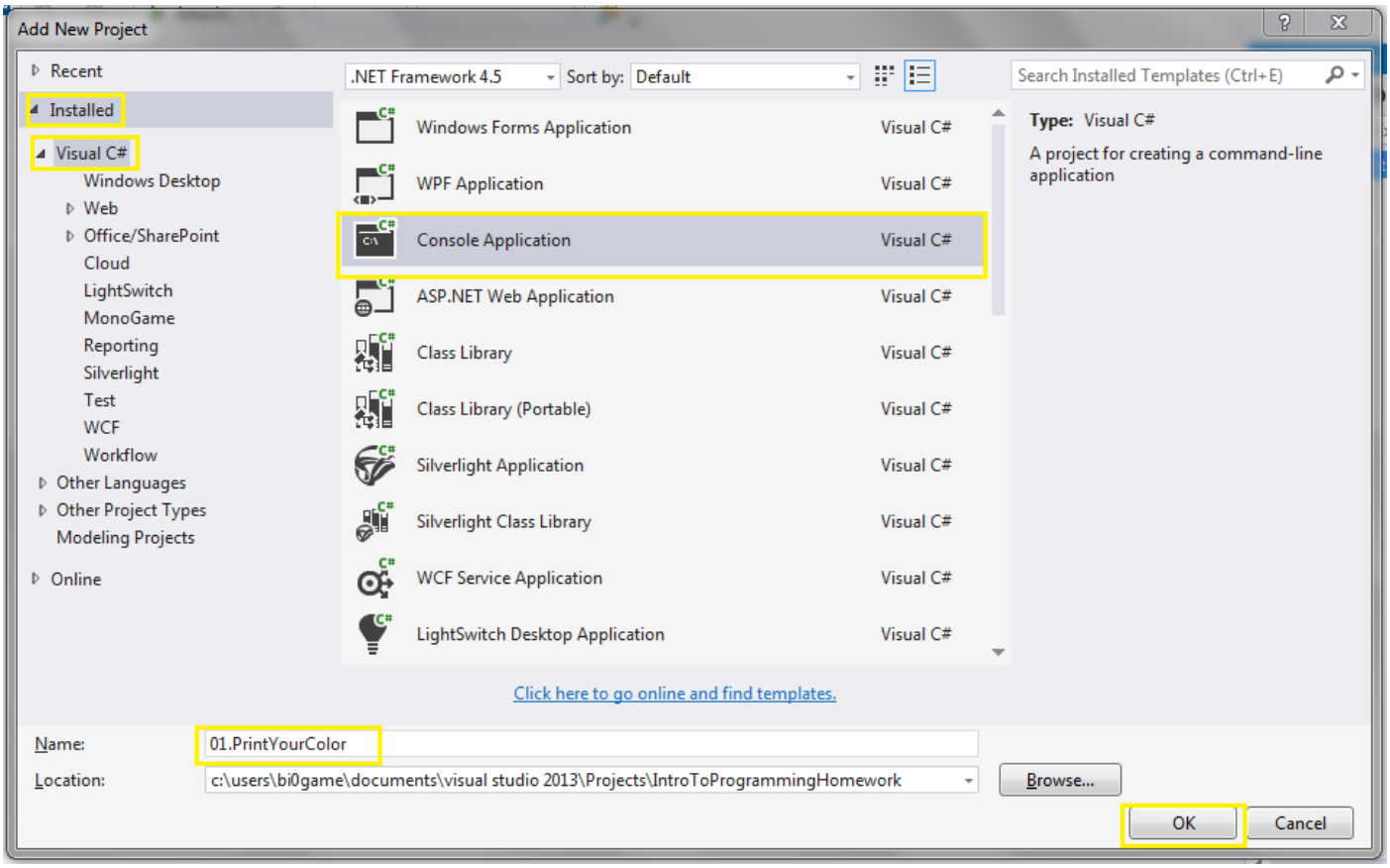




Problem 2. Adding Projects To Solution

From the top menu select **File -> Add -> New Project**. Then from the menu to the left choose **Installed -> Visual C#** and then select from the middle menu select **Console Application**, **name it appropriately** and click **OK**.





Problem 3. Fix The Auto-Generated Code

After you have added your new project to the solution, you need to **fix some** of the automatically generated **code** done by Visual Studio. First you need to **remove** some of the **libraries** that you **do not need**. Then you need to **remove** the **namespace** as you do not need it as well. Next thing is to **rename** the **class** so that it has a **descriptive name** (as shown in the picture) and **remove** the "**string[] args**" in the **parentheses**. Finally you need to **rename** the **.cs file** in the project to **match** the **class' name**.

Problem 4. Write Some Code

You need to write some code that prints out your favorite color.

1. Between the {} parentheses of the Main method you need to write the Console.WriteLine(); method.
2. Now you need to write your favorite color in double-quotes and put it in the WriteLine method.
3. Finally you need to press Ctrl + F5 to build and run your program

```
using System;
0 references
class PrintYourColor
{
    0 references
    static void Main()
    {
        Console.WriteLine("Red");
    }
}
```

02.Primitive Data Types and Variables

Problem 5. Practice Integer Numbers

Create a new C# project and create a program that **assigns integer values** to **variables**. Be sure that each **value** is stored in the **correct variable type** (try to find the most suitable variable type in order to save memory). Finally you need to **print** all variables to the console.

Values	Output
-100 128 -3540 64876 2147483648 -1141583228 -1223372036854775808	-100 128 -3540 64876 2147483648 -1141583228 -1223372036854775808

Problem 6. Practice Floating Point Numbers

Create a new C# project and create a program that **assigns floating point values** to **variables**. Be sure that each **value** is stored in the **correct variable type** (try to find the most suitable variable type in order to save memory). Finally you need to **print** all variables to the console.

Values	Output
3.141592653589793238 1.60217657 7.81842619745845552165353423 41	**3.141592653589793238** 1.60217657 7.81842619745845552165353423 41

Problem 7. Practice Characters and Strings

Create a new C# project and create a program that **assigns character and string values** to **variables**. Be sure that each **value** is stored in the **correct variable**.

Values
Software University **B** **y** **e** **I love programming**

03.Exercises: Operators Expressions and Statements

Problem 8. Average

Write a program that finds the **average** of the **sum of 3** numbers.

Hints:

1. Declare ***four*** variables (*a*, *b*, *c* and *average*).
2. **Read** the user input from the console. (int.Parse(Console.ReadLine());).
3. **Calculate** the **average value** of the variables by the **formulae** ($average = \frac{a+b+c}{3}$).
4. **Print** the result on the console (Console.WriteLine(average));).

a	b	c	Average
45	41	20	35.33333
22	52	60	44.66667

Problem 9. Trapezoid

Write a program that finds the **area** of a trapezoid, given the base sides **a**, **b** and height **h**.

Hints:

1. Declare ***four*** variables (*a*, *b*, *h* and *area*).
2. **Read** the user input from the console. (int.Parse(Console.ReadLine());).
3. **Calculate** the **area of the trapezoid** by the **formulae** ($area = \frac{a+b}{2} * h$).
4. **Print** the result on the console (Console.WriteLine(area));).

a	b	h	Area
5	2	4	14
8.5	4.4	2	12.9

Problem 10. Last Digit

Write a program that prints the last digit of a number **n**.

Hints:

1. Declare ***two*** variables (*n* and *lastDigit*).
2. **Read** the user input from the console. (int.Parse(Console.ReadLine());).
3. **Find** the **last digit** of the **number** by the **formulae** ($lastDigit = n\ mod(10)$). The word ***mod*** means modular division (or the operator **%** in C#).
4. **Print** the result on the console (Console.WriteLine(lastDigit));).

n	Result
21	1

139	9
4	4

Problem 11. N-th Digit

Write a program that prints the **n**-th digit of a number (from right to left). If no such digit exists, print a dash "**-**".

Hints:

1. Declare ***three*** variables (*number*, *n* and *nDigit*).
2. **Read** the user input from the console. (int.Parse(Console.ReadLine());).
3. **Find** the **n-th digit** of the **number** by using the **formulae** ($nDigit = \frac{number}{10^{n-1}} mod(10)$). The word ***mod*** means modular division (or the operator **%** in C#).
4. **Print** the result on the console (Console.WriteLine(area));).

Number	n	Result
2174	3	1
169	2	6
46	4	-

Problem 12. Big and Odd

Write a program that that prints if the number is both **greater than 20** and **odd**.

Hints:

1. Declare ***two*** variables (*n* and *result*).
2. **Read** the user input from the console. (int.Parse(Console.ReadLine());).
3. **Check** if the input **number** is **greater** than **20** and **odd** by using the **logical operators**:
    a. **>** or **<** checks if the **value on the left** of the operator is **greater/less** than the **value on the right** side of the operator;
    b. Using the **formulae** ($number\ mod(10) == 1$) you check whether the entered number **is odd**. The word ***mod*** means modular division (or the operator **%** in C#);
    c. **&&** checks if the **left expression AND** the **right expression** both have **true** values;
    d. **Save** the **result** of the verification in the ***result*** variable;
4. **Print** the result on the console (Console.WriteLine(result));).

n	Result
63	true
17	false
22	false
23	true

Problem 13. Pure Divisor

Write a program that prints if a number is **divided** by 9, 11 or 13 **without remainder**.

Hints:

1. Declare ***two*** variables (*n* and *result*).
2. **Read** the user input from the console. (int.Parse(Console.ReadLine());).
3. **Check** if the input **number** is **divided** by **9**, **11** or **13** using the **logical operators**:
    a. Using the **formulae:**
    $(number\ mod(9) == 0$ OR $number\ mod(11) == 0$ OR $number\ mod(13) == 0)$
    you check whether the entered number **is divided** by the given **constants without remainder**. The word ***mod*** means modular division (or the operator **%** in C#);
    b. **||** checks if the **left expression OR** the **right expression** have **a true** value. If **only one** has a **true** value the result is **true**;
    c. **Save** the **result** of the verification in the ***result*** variable;
4. **Print** the result on the console (Console.WriteLine(result));).

n	Result
121	true
1263	false
26	true
23	false
81	true
1287	true

04. Exercises: Conditional Statements

Problem 14. Biggest of three

Write a program that finds the **biggest number** from given 3 numbers. Print the result on the console.

Hints:

5. Declare ***three*** variables (*a*, *b*, *c*).
6. **Read** the user input from the console.
7. **Find** the **biggest** number by using conditional **if** statements.
8. **Print** the result on the console.

a	b	c	Max
45	41	20	45

22	52	60	60

Problem 15. Sign of product

Write a program that **finds** the **sign** of the **product** of **three** real numbers. Use only **if** conditional statements.

Hints:

5. Declare ***four*** variables (*a*, *b*, *c* and *product*).
6. **Read** the user input from the console.
7. Use if statements to check the sign of the product.
    a. Check the sign of each number.
    b. If there are 1 or 3 negative numbers – the product is negative.
    c. If there are 0 or 2 negative numbers – the product is positive.
8. **Print** the result on the console.

a	b	c	Sign
5	2	4	Positive
8.5	4.4	-2	Negative
2.7	-0.1	-9	Positive

Problem 16. Number as Day of Week

Write a program that asks for a **digit** (1-7), and depending on the input, **shows** the corresponding **day of week** as a **word** (in English). Print "not valid" in case of invalid inut. Use a **switch** statement.

Hints:

5. Declare one variable (*n*).
6. **Read** the user input from the console. (int.Parse(Console.ReadLine());).
7. Using a switch-case statement declare the possible values:
    a. Numbers in the range [1…7] are corresponding to the day of week. (E.g. 1 -> Monday).
    b. Other numbers or not valid inputs should be "not valid".
8. **Print** the result on the console.

d	result
2	Tuesday
1	Monday
0	not valid
5	Friday
-0.1	not valid

hi	not valid
7	Sunday
10	not valid

05. Exercises: Loops

Problem 17. Calculate N!

Write a program that finds the **factorial** of a given **number**. Print the result on the console.

Hints:

9. **Declare** two variables (*n, factorial*).
10. **Assign** the ***factorial*** variable the value of **1**.
11. **Read** the user input from the console.
12. **Calculate factorial** of the given number:
    a. Use any kind of loop that starts from **2** and ends at the **number** given in the user input **including**. E.g. *for(int i = 2; i <= n; i++)…*
    b. At each iteration of the loop calculate the product of ***factorial*** and the iterator variable ***i.*** E.g.
    *factorial = factorial * i;*
13. **Print** the result on the console.

n	factorial
5	120
6	720