/\*

Комментарий

многострочный

\*/

// комментарий по строке

// комментарий по строке Ctrl+E,C

/// <summary>

/// Название подписание

/// </summary>

/// <param name="args">параметры запуска</param> - зачем параметры запуска  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

WriteLine("Hello World!"); показать

ReadKey(); получить

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

using System;  
using static System.Console;  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Переменная (имя) – идентификатор, «контейнер» хранения данных.

var a = 28;  
var b = 12,2;  
var c = “Текст”;

Числовые типы:  
  
Целые числа со знаком.  
sbyte int8 = 2; // от -128 до +127  
short int16 = 1; // от -32768 до +32767;  
int int32 = 0; // от - 2 147 483 648 до 2 147 483 647  
long int64 = -1; // от -9 223 372 036 854 775 808 до 9 223 372 036 854 775 807  
  
Целые числа без знака.  
byte uInt = 0; // от 0 до 255  
ushort uInt16 = 1; // от 0 до 65535  
uint uInt32 = 2; // от 0 до 4 294 967 296  
ulong uInt64 = 3; // от 0 до 18 446 744 073 709 551 616  
  
int \_numb = 1\_999\_222; для удобсва  
  
Числа с плавающей точкой.  
float floatD = 3.14F; // точность 7 цифр 1.1234567  
double doubleD = 3.14D; // точность 15 цифр 1.\*\*\*\*\*\*\*\*\*\*\*\* (d в конце необязательный)  
decimal decimalD = 3.14M; // точность 28 цифр 1.\*\*\*\*\*\*\*\*\*\*\*  
  
Console.WriteLine(floatVariable);  
Console.WriteLine(doubleVariable);  
Console.WriteLine(decimalVariable);  
Символьный - ‘a’ юникод символ  
char VarChar1 = ‘a’; Console.WriteLine(VarChar1);  
  
Логический – true, false

Bool flag = true;  
flag = false;  
  
Строковые типы – “Слова“  
String text = “текст”;

Контактенация (“Возвраст: “ + aslld)

String name = "Александр";

String nameFirst = "Лазарев";

Console.WriteLine("Меня зовут {0} можно писать внутри кавычек {1}",

name,

nameFirst);

String name = "Александр";

String nameFirst = "Лазарев";

String pattern = "Меня зовут {0} можно писать внутри кавычек {1}";

Console.WriteLine(pattern,

name,

nameFirst);

String fname = "Марина";

String lname = "Лазарева";

Console.WriteLine (pattern,

fname,

lname);

pattern пишем один раз, а используем в разных местах с разными значениями. Escape последовательности.

\n новая строка

\r возврат каретки

\t горизонтальная табуляция

\’ одиночная кавычка

\” двойная кавычка

\\ обратная косая черта

\? Литерал вопросительного знака

Интерполяция строк. Вывод строк значений переменных. $ перед кавычками.  
Можно подставить переменные {} напрямую, не заморачи С контактенацией.

int a = 3;

int b = 4;

int c = 5;

Console.WriteLine($"{a} + {b} = {c}");

String name = "Александр", lastName = "Лазарев";

String name1 = "Марина", lastName1 = "Лазарева";

Console.WriteLine($"{name} {lastName} \n{name1} {lastName1}");

Console.WriteLine($"{name1} {lastName1}");   
  
 Console.WriteLine($"{name,10} {lastName,14}");

Console.WriteLine($"{name1,10} {lastName1,14}");  
  
Числа количество отведенных символов под это значение.

Числовые типы, количество чисел после запятой и разделение пробелами

double d = 1233.1231231;

String dFormated = d.ToString("## ##.## ##");

Console.WriteLine(dFormated);

//Вывод по заданому формату ToString

Console.WriteLine("{0:00 00.00 00 0}", d);

//Вывод по строке и формату через :

Console.WriteLine(d.ToString("## ##.## ## #"));  
 //Вывод по заданому формату ToString

Дата (в зависимости от настроек языкового форматы могут различаться)  
Вывод даты в разных орматах  
  
 var date = new DateTime(2020, 02, 11, 01, 22, 58);

Console.WriteLine(date);

Console.WriteLine($"{date:HH:mm}");

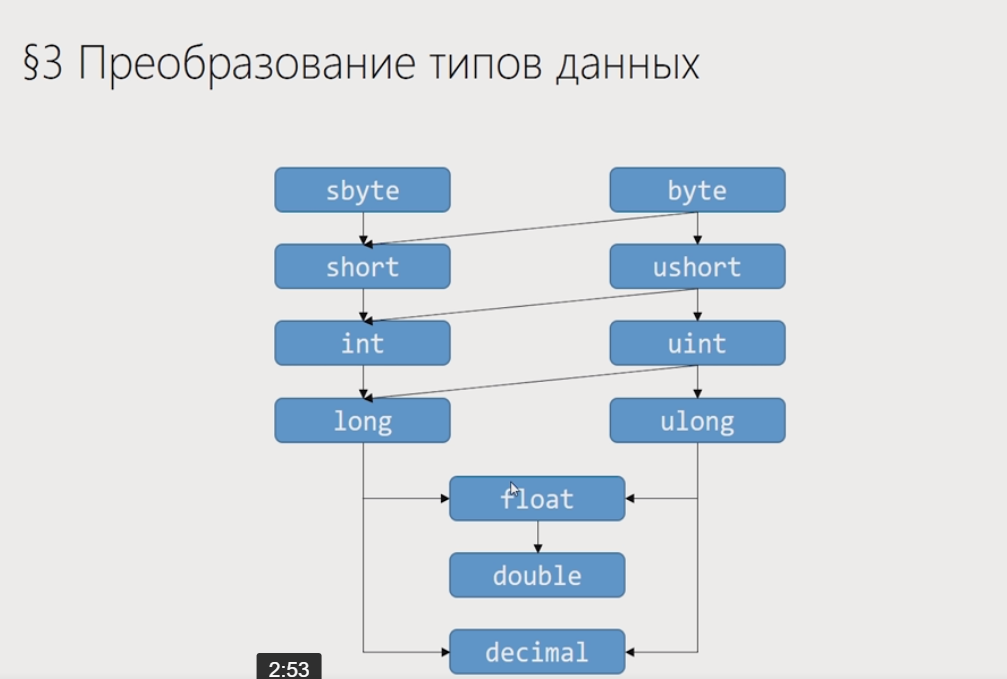
Console.WriteLine($"{date:yyyy-MM-dd}");

Console.WriteLine($"{date:dd.MM.yyy}");

------------------------------------------------------------------------------------

Числовые типы:  
  
Целые числа со знаком.  
sbyte int8 = 2; // от -128 до +127  
short int16 = 1; // от -32768 до +32767;  
int int32 = 0; // от - 2 147 483 648 до 2 147 483 647  
long int64 = -1; // от -9 223 372 036 854 775 808 до 9 223 372 036 854 775 807  
  
Целые числа без знака.  
byte uInt = 0; // от 0 до 255  
ushort uInt16 = 1; // от 0 до 65535  
uint uInt32 = 2; // от 0 до 4 294 967 296  
ulong uInt64 = 3; // от 0 до 18 446 744 073 709 551 616  
  
int \_numb = 1\_999\_222; для удобства  
  
Числа с плавающей точкой.  
float floatD = 3.14F; // точность 7 цифр 1.1234567  
double doubleD = 3.14D; // точность 15 цифр 1.\*\*\*\*\*\*\*\*\*\*\*\* (d в конце необязательный)  
decimal decimalD = 3.14M; // точность 28 цифр 1.\*\*\*\*\*\*\*\*\*\*\*

Что во что можно преобразовать.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Неявное преобразование, так как число max число байт, меньше max числа int

Меньшие типы чисел, в большие.  
**Неявные преобразования**. Специальный синтаксис не требуется, так как преобразование всегда завершается успешно и данные не будут потеряны. Примеры включают преобразования из меньших в большие целочисленные типы и преобразования из производных классов в базовые классы.

byte b = 100;

int a = b;

Console.WriteLine(a);

double d = a;

Console.WriteLine(d);

\_\_\_\_\_\_

byte byteValue = 1;

short shortValue = byteValue;

int intValue = shortValue;

long longValue = intValue;

double doubleValue = longValue;

char c = 'k';

int d = c;

с таблицы кодировок.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Явное преобразование. (приведение типов) Ответственность лежит на нас, данные могут быть некорректны и потеряны.

int i = 330;

byte j = i; - ошибка

int i = 330;

byte j = (byte)i; = 74

\_\_\_\_\_\_\_

double d = 1.77;

int h = (int)d;

Console.WriteLine(h);

int f = (byte)d; явное присвоение типа double к байт, int к byte.

Console.WriteLine(f);

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Преобразования с использованием вспомогательных классов. Чтобы выполнить преобразование между несовместимыми типами.**   
Метод Parse встроенный внутри числовых типов int.Parse(String); double.Parse(String);

string inputString = "2029";

int intValue = int.Parse(inputString);

double doubleValue = double.Parse(inputString);

ulong ulongValue = ulong.Parse(inputString);

//byte byteValue = byte.Parse(inputString);

Console.WriteLine($"{intValue} {doubleValue} {ulongValue} byte тут не будет работать)))");

\_\_\_\_\_\_\_\_\_\_\_

Метод Convert встроенный внутри любой тип в любые типы.  
int – ToInt32 – синоним .NET посмотреть можно через F1, выделив тип данных.

string inputString = "2029";

int intValue = Convert.ToInt32(inputString);

long longValue = Convert.ToInt32(intValue); - приведение стринг-инт-лонг

ulong ulongValue = Convert.ToUInt64(inputString);

double doubleValue = Convert.ToDouble(longValue);

Console.WriteLine($"{intValue} {longValue} {ulongValue}");  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Посмотреть названия чисел .Net можно через F1 справка.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Арифметические операции   
+ - \* /**

int valueA = 2\_233\_222; int valueB = 778;

int valueC = valueA + valueB;

Console.WriteLine(valueC);

long valueL = 2332; byte valueR = 22;

var valueS = valueL + valueR; - VAR универсальное, сам подберет.

Console.WriteLine(valueS);

Деление. Целое число. Остаток остается. 59/8 = 7+3

\_\_\_\_\_\_\_\_\_\_\_\_

Остаток от деления %.

a = b \* q + r

Console.WriteLine("Деление в целых числах. Остаток");

Console.WriteLine($"5 / 2 = {5 / 2}"); //2

Console.WriteLine($"5 % 2 = {5 % 2}"); //1

Console.WriteLine($"59 % 8 = {59 % 8}"); //3

Console.WriteLine($"21 % 4 = {21 % 4}"); //1

\_\_\_\_\_\_\_\_\_\_\_\_

Сокращение арифметика

int a = 12;

int b = 13;

int c = a + b;

Console.WriteLine(a); //12

a = a + b;

a += b;

a -= b;

a /= b;

a %= b;

Console.WriteLine(a); //25

\_\_\_\_\_\_\_\_\_\_\_\_

++ --

int a = 10;

a += 20; //30

a += 1; //31

a++; //постфиксный инкремент +1

++a; //префиксный инкремент +1

\_\_\_\_\_\_\_\_\_\_\_\_

// Приоритет операций/

// Унарные операции – 10 и +10 одно и то же.

// Унарный минус –

int a = 10;

a = -a; // a = -1 \* a, -10

// Префиксный инкремент ++a

int a = 10;

Console.WriteLine(a++); //10

Console.WriteLine(a); //11

Console.WriteLine(++a); //12

Console.WriteLine(a); //12

// %, /, \*

// +, -

// Постфиксный инкремент a++

// () – скобки имеют приоритет.

------------------------------------------------------------------------------------------------------------------------------------------

Логические операции. bool true (1), false (0)   
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Инверсия / Отрицание / Логическое «НЕ» / !

bool variable1 = true;

bool variable2 = false;

Console.WriteLine($"variable1 = {variable1} !variable1 = {!variable1}");

Console.WriteLine($"variable1 = {variable2} !variable1 = {!variable2}");  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Конъюнкция / Логическое умножение / Логическое «И» / &&  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Дизъюнкция / Логическое сложение / Логическое «ИЛИ» / ||  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Разделительная дизъюнкция / Строгая дизъюнкция / Исключающее «ИЛИ» / ^  
Когда одна из нескольких true, тогда true, если одинаковы – false.  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
 char c = 'f';

bool flag1 = 4 > 5; Console.WriteLine(flag1); //false

bool flag2 = 1 < 3; Console.WriteLine(flag2); //true

bool flag3 = c == 'f'; Console.WriteLine(flag3); //true

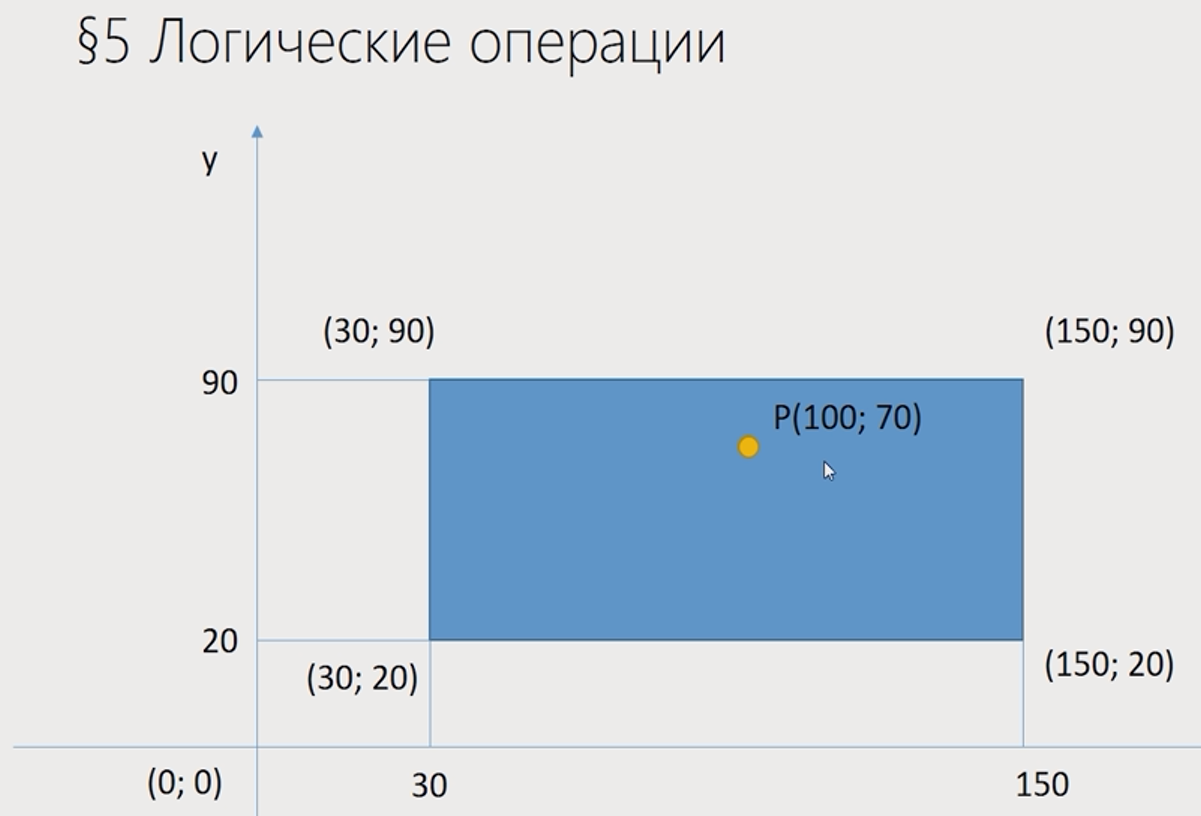
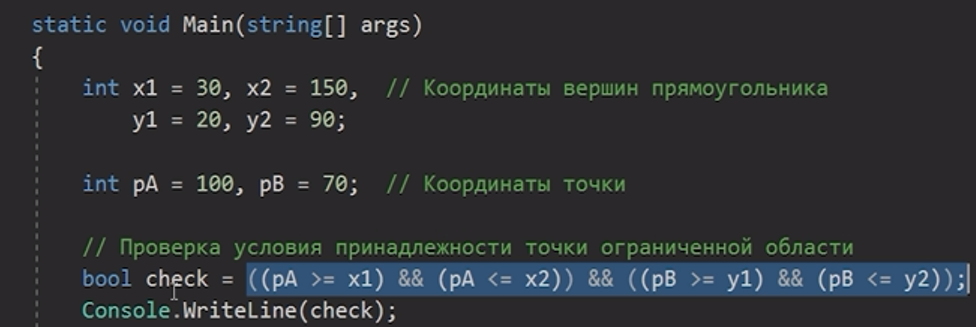
bool flag4 = 28 == 28; Console.WriteLine(flag4); //true

bool flag5 = 28 >= 20; Console.WriteLine(flag5); //true

bool flag6 = 9 <= 9; Console.WriteLine(flag6); //true

bool flag7 = 28 != 90; Console.WriteLine(flag7); //true  
 string s = "C#";

bool flag8 = 28 != 90 && s == "C#"; Console.WriteLine(flag8); //true  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Мы научились до этого момента выводить с помощью конкатенации строк, форматированного вывода и интерполяции строк – константы, дальше учимся использовать вводные данные.   
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Условный оператор.   
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Оператор Console.WriteLine и Console.ReadLine

string name;

byte age;

uint costs;

Console.WriteLine("Вводи имя:");

name = Console.ReadLine();

Console.WriteLine("Вводи возраст:");

age = byte.Parse(Console.ReadLine()); //обязательно парсим стринг в байт

Console.WriteLine("Вводи сколько тратишь рублей в месяц:");

costs = Convert.ToUInt32(Console.ReadLine()); //конвертируем стринг в ushort

Console.WriteLine($"Ну и имя: {name}\nНу вот уже {age} еще немного и старость\nТраты: {costs} рублей - это капец дохера для тебя {name}");  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Генерация автоматическая.

Random randomize = new Random(); //Создание переменной randomize для получения случайных чисел

Console.WriteLine($"int.Maxvalue = {int.MaxValue}"); //получение максимального числа int

Console.WriteLine($"int.Maxvalue = {int.MinValue}"); //получение минимального числа int

int randomIntResult = randomize.Next(); // Возвращает случайное неотриц целое число от 0 до maxInt.

Console.WriteLine($"Пример 1. randomIntResult = {randomIntResult}");

randomIntResult = randomize.Next(100); // Возвращает случайное неотриц целое число, от 0 до 99

Console.WriteLine($"Пример 2. randomIntResult = {randomIntResult}");

randomIntResult = randomize.Next(10, 100); // Возвращает от 10 до 99

Console.WriteLine($"Пример 3. randomIntResult = {randomIntResult}");

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

double randomDoubleResult = randomize.NextDouble() \* 10; // Возвращает случайное число с плав запятой >= 0.0 и <1.0, не включая 1.0  
  
 // double randomDoubleResult = randomize.NextDouble() + randomize.Next(100); по действиям

Console.WriteLine($"Пример 2. randomIntResult = {randomDoubleResult}");

------------------

Оператор ветвления.  
Для выполнения различных блоков кода на основе разных условий.

if – истина, условие может включать в себя любое сравнение и логические операторы (bool).

Random rand = new Random();

int x = rand.Next(3); //0, 1 или 2

Console.WriteLine($"Получено значение х = {x}");

if (x < 2) // tab 2 раза

{

Console.WriteLine("x равен 0 или 1");

}

\_\_\_\_\_\_\_\_\_\_\_\_\_

If, else  
Можно использовать много if’ов, они все будут проверены по очереди, пока предыдущие условия ложные.

if (x < 2) // tab 2 раза

{

Console.WriteLine("x равен 0 или 1");

}

else

{

Console.WriteLine("x равен 2");

}

\_\_\_\_\_\_\_\_\_\_\_\_

if – else – if каскад условий  
условие да или нет, условие да или нет, если не подошло предыдущее

if (x < 2)

{

Console.WriteLine("x равен 0 или 1");

}

else if (x == 2)

{

Console.WriteLine("x равен 2");

}  
 else

{

Console.WriteLine("x равен 2");

}

\_\_\_\_\_\_\_\_\_\_\_

if-else-if каскад условий – более правильно

if (x < 2)

{

Console.WriteLine("x равен 0 или 1");

}

else

{

if (true)

{

Console.WriteLine("true");

}

else

{

Console.WriteLine("false");

}

}

Random rand = new Random();

int temp = rand.Next(25); //0, 1 или 2

Console.WriteLine($"Температура {temp} С");

if (temp < 15)

{

Console.WriteLine("Холодно, возьми куртку");

}

else

{

Console.WriteLine("Теплооооооооооо");

}

int firstNumber = rand.Next(-10,11); //заказчик просил от -10 до 10

int secondNumber = rand.Next(-10,11); //заказчик просил от -10 до 10

if (firstNumber > secondNumber)

{

Console.WriteLine($"Максимальное число {firstNumber}");

Console.WriteLine($"Минимальное число {secondNumber}");

}

else

{

Console.WriteLine($"Минимальное число {firstNumber}");

Console.WriteLine($"Максимальное число {secondNumber}");

}

bool flag = firstNumber > secondNumber;

if (flag)

{

Console.WriteLine($"Максимальное число {firstNumber}");

Console.WriteLine($"Минимальное число {secondNumber}");

}

else

{

Console.WriteLine($"Минимальное число {firstNumber}");

Console.WriteLine($"Максимальное число {secondNumber}");

}

МОЖНО СОХРАНЯТЬ РЕЗУЛЬТАТЫ ВЫЧИСЛЕНИЯ, ЕСЛИ НУЖНО.Если числа равно добавить

Console.Write("Введите первое число: ");

int firstNumber = int.Parse(Console.ReadLine());

Console.Write("Введите второе число: ");

int secondNumber = int.Parse(Console.ReadLine());

Console.Write("ВВедите день недели: ");

int firstNumber = int.Parse(Console.ReadLine());

if (firstNumber == 1) Console.WriteLine("Понеденльик");

else if (firstNumber == 2) Console.WriteLine("Вторник");

else if (firstNumber == 3) Console.WriteLine("Среда");

Тернарный оператор  
  
переменная = (УСЛОВИЕ) ? true : false   
 Random r = new Random();

double x = r.NextDouble();

int result = (x > 0) ? 1 : 0;

Console.WriteLine(result);

Console.WriteLine(x > 0 ? 1 : 0);

\_\_\_\_\_\_\_\_\_

int maxValue = (a > b) && (a > c) ? a : (b > c) ? b : c;  
\_\_\_\_\_\_\_\_\_

2 random числа

bool f = a > b;

string outputInt = f ? "да" : "нет";  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оператор выборки  
Switch (переменная для сравнения)  
{case 1: “ss” break }break

Console.WriteLine("Ввведите номер дня: ");

int dayIndex = int.Parse(Console.ReadLine()); //переводим в инт

switch (dayIndex)

{

case 1:

Console.WriteLine("Понедельник");

break;

case 2:

Console.WriteLine("Вторник");

break;

case 3:

Console.WriteLine("Среда");

break;

default:

Console.WriteLine("Другой день");

break;

}

default == else  
\_\_\_\_\_\_\_\_\_\_\_\_

Switch при сравнении числа придется писать все строки

Case 0,1,2,3,4 будет равен четвертому значению.  
  
Case 5,6,7,8,9, будет равен девятому значению.  
  
 switch (value)

{

case 0:

case 1:

case 2:

case 3:

case 4: Console.WriteLine("0 <= value < 5");

case 5:

case 6:

case 7:

case 8:

case 9: Console.WriteLine("5 <= value < 9");

default:

break;

}

ЦИКЛЫ. Оптимизация рутинных задач.

Повторения блока кода, может быть назначено, либо бесконечно. Кусочек инструкций – ТЕЛО ЦИКЛА.

Выполнение тела цикла – ИТЕРАЦИЯ цикла.  
\_\_\_\_\_\_\_\_\_\_\_\_\_

for

счетчик условие прирощение

for (int I = 0; I < 10; i++)

{

//тело цикла

}

Random rand = new Random();

//Console.WriteLine("Введите число");

//int index = int.Parse(Console.ReadLine());

int index = 10;

for (int i = 0; i < index; i++) // от 0 до 10

{

Console.Write($"{i} ");

}

for (int j = 0; j < index; j++) //от 10 до 0

{

Console.Write($"{10 - j} ");

}

for(double s = -1; s <= 1; s += 0.25) //от -1 до 1 шаг 0.25

{

Console.Write($"{s} ");

}

Random rand = new Random();

//Console.WriteLine("Введите число");

//int index = int.Parse(Console.ReadLine());

for (int i = 0; i < 100; i++) //генерация случайных 100 чисел от -8 до 12

{

Console.Write($"{rand.Next(-8,12)} ");

}

double value; // средний балл предметов

double average = 0; // сумма всех баллов

for (int i = 1; i <= 5; i++) // начало с какого; сколько предметов; 1+

{

Console.WriteLine($"Введите {i}-й результат экзамена: ");

value = double.Parse(Console.ReadLine());

average += value;

}

Console.WriteLine($"Среднее = {average / 10}");

}

for может быть с условиями внутри, или вообще без условий for (; ; )

break; - выбрасывает с цикла.   
Угадай-ка  
 Random rand = new Random();

Console.WriteLine("Угадывай-ка число");

int number = rand.Next(1, 101);

int count = 0;

int usernumber;

for (; ; )

{

Console.Write("Вводи число: ");

usernumber = Convert.ToInt32(Console.ReadLine());

count++;

if (usernumber < number)

{

Console.WriteLine("Твое число меньше");

}

else if (usernumber > number)

{

Console.WriteLine("Твое число больше");

}

else

{

Console.WriteLine($"Молодец, угадано за {count} попыток");

break;

}

While

Переменная (переменная с условием)

While (условие) {

Тело цикла

}

int i = 0;

while (i < 15) {

Console.WriteLine(i);

i++;

}

\_\_\_\_\_\_\_\_

int i = 0;

while (i < 26) {

Console.WriteLine(i);

i += 3;

}

//сгенерировать 10 случайных чисел от -8 до 100

Random rand = new Random();

int numb = 0;

while (numb < 10 )

{

Console.WriteLine($"{rand.Next(-8, 101)}");

numb++;

}

//Угадай-ка

Random rand = new Random();

int numb = rand.Next(1,81);

int numbuser = 0 ;

int count = 0;

Console.WriteLine("Введи число:");

while (numb != numbuser)

{

numbuser = Convert.ToInt32(Console.ReadLine());

if (numb < numbuser)

{

Console.WriteLine("Нужно ввести меньше ");

}

else {

Console.WriteLine("Нужно ввести больше ");

}

count++;

}

Console.WriteLine($"Молодец, угадал. Попыток: {count}");

}

Можно так, а можно изменить условия цикла, внутри цикла сделать еще if else и через break выходить

do-while

Цикл с пост условием. Условие проверяется только после выполнения цикла. Т.е. как минимум один прогон.  
Сначала действие потом проверка.  
  
bool condition = false;

do

{

Тело цикла

} while (condition);

\_\_\_\_\_\_\_\_\_

int i = 1;

do

{

Console.WriteLine($"Число: {i}");

i++;

} while (i < 11);

Continue;

for (int i = 1; i < 10;i++)

{

if (i < 5) continue; //начинает цикл заново если условие подходит

Console.WriteLine(i);

}

System.Diagnostics.Debug

МАССИВЫ  
Массив – структура данных, хранящая набор значения (элементов массива), идентифицируемых по индексу или набору индексов.  
  
Массив данных имеет одинаковый тип данных.   
int[] x; объявление массива. Тип[] имя массива.

Массив объявляется с ключевым словом **new,** за которым идет информация о типе данных и количестве элементов массива. Это фиксированное количество, которое может содержать массив. (умолчания числа – нули, bool – false и т.д.)

int[] x;

x = new int[3];

double[] array = new double[5];

x[0] = 1;

x[1] = 1;

x[2] = 3;

array[0] = 2;

array[1] = 3;

array[2] = 4;

array[3] = 5;

array[4] = 6;

int[] birthdayCollection = new int[8] { 2, 6, 8, 5, 7, 4, 5, 6 }; // указать количество и описать их сразу

int[] newBirthdayCollection = new int[] { 2, 6, 2222, 33336 }; // не указывать количество и описать их сразу

int[] testCokkection = { 02, 22, 1423 }; // можно new не указывать и количество

Console.WriteLine("Сформировать масси из N случайных чисел от -5 до 5");

Console.Write("Введите n: "); //приглашение к вводу числа

int n = int.Parse(Console.ReadLine()); //ввод числа с клв

Random r = new Random(); // инициализация генератора псевдослучайных чисел

int[] numbs = new int[n]; //объявляем массив, число вводит пользователь

for (int i = 0; i < numbs.Length; i++)

{

numbs[i] = r.Next(-5, 6); //заполнение массива

Console.Write($"{numbs[i]} "); //вывод массива

}

Console.WriteLine("Сформировать масси из N случайных чисел 0 до 9, с подсчетом четных чисел.");

Console.Write("Введите n: "); //приглашение к вводу числа

int n = int.Parse(Console.ReadLine()); //ввод числа с клв

Random r = new Random(); // инициализация генератора псевдослучайных чисел

int[] numbs = new int[n];

int evenElement = 0;

for (int i = 0; i < numbs.Length; i++) //Length длина массива.

{

numbs[i] = r.Next(10);

if (numbs[i] %2 == 0) evenElement++; //Если число четное, то +1

Console.Write($"{numbs[i]} ");

}

Console.WriteLine($"\nКоличество четных чисел = {evenElement}");

Console.WriteLine("Сколько раз встречается число из массива 10 чисел");

Random r = new Random();

int[] numbs = new int[10];

int findNumbers = 0; //результат поиска

for (int i = 0; i < numbs.Length; i++) //Length длина массива.

{

numbs[i] = r.Next(10);

Console.Write($"{numbs[i]} ");

}

Console.Write($"\nВведите число, которая будет искать: ");

int number = int.Parse(Console.ReadLine());

for (int i = 0; i < numbs.Length; i++)

{

if (number == numbs[i]) findNumbers++;

}

Console.WriteLine($"\nВаше число {number} встречается {findNumbers} раз");

Сортировка.  
 Console.WriteLine("Массив из n элементов, состоит из 0 и 1. \nСортировать 0 в начало. 1 в конец.");

Console.WriteLine("Количество элементов: ");

int n = int.Parse(Console.ReadLine());

int[] numbs = new int[n];

Random random = new Random();

Console.Write("Исходный массив: ");

for (int i = 0; i < numbs.Length; i++)

{

numbs[i] = random.Next(0,2);

Console.Write(numbs[i] + " ");

}

int left0 = 0;

int right1 = numbs.Length - 1; //последний элемент всегда - 1

bool flag = true; //true - увеличивать позицию left, false уменьшает позицию right

while (left0 < right1) //пока голова меньше хвоста

{

if (flag)

{

if (numbs[left0] == 1) flag = false; //работаем с левым элементом

else left0++;

}

else

{

if (numbs[right1] == 0) flag = true; //работаем с хвостом

else right1--;

}

if (numbs[left0] == 1 && numbs[right1] == 0) //если слева 1, справа 0, то меняем

{

int temp = numbs[left0];

numbs[left0] = numbs[right1]; //меняем местами

numbs[right1] = temp;

left0++; //сдвигаем голову

right1--; //сдвигаем хвост

}

}

Console.WriteLine($"\nВывод массива: ");

for (int i = 0; i < numbs.Length; i++)

{

Console.Write(numbs[i] + " ");

}

Array.sort(); - сортировка данных.

Console.WriteLine("Массив из n элементов, состоит из 0 и 1. \nСортировать 0 в начало. 1 в конец.");

Console.WriteLine("Количество элементов: ");

int n = int.Parse(Console.ReadLine());

int[] numbs = new int[n];

Random random = new Random();

Console.Write("Исходный массив: ");

for (int i = 0; i < numbs.Length; i++)

{

numbs[i] = random.Next(0,2);

Console.Write(numbs[i] + " ");

}

Array.Sort(numbs);

Console.WriteLine($"\nВывод массива: ");

for (int i = 0; i < numbs.Length; i++)

{

Console.Write(numbs[i] + " ");

}

Array

var data = new int[] { 1, 2, 5, 4, 5 };

Console.WriteLine($"data.Length = {data.Length}"); //число элементов во всех измерениях массива

int result = Array.IndexOf(data, 5); //выполняет поиск указанного объекта и показывает позицию первого вхождения

Console.WriteLine(result);

int lastResult = Array.LastIndexOf(data, 5);

Console.WriteLine(lastResult);

int rank = data.Rank; //число измерений массива

Array.Sort(data); //сортирует элементы во всем массиве

Array.Reverse(data); //сортирует от большего к меньшему

Array.Clear(data, 3, 2); //задает значение по умолчанию, где 3-3элемент (4). 2 количество от него

int[] copyData = new int[20];

Array.ConstrainedCopy(data, 1, copyData, 20, 20); //копирует массив с одного в другой.(исходный, с какого копировать, приемник, с какой вставлять, сколько копировать), если неуспешно, то все отменит

Array.Copy(data, 0, copyData, 20, 20);

Цикл, в цикле. Таблица умножения.  
  
 for (int i = 2; i < 10; i++)

{

for (int j = 1; j <= 10; j++)

{

Console.WriteLine($"{i}x{j}={j \* i} ");

}

Console.WriteLine();

}

//Показать всевозможные трехбуквенные слова из алфавита S K I L

// 4\*4\*4 = 4^3

char[] symbols = { 'S', 'K', 'I', 'L' };

int count = 1;

for (int i = 0; i < symbols.Length; i++)

{

for (int j = 0; i < symbols.Length; j++)

{

for (int k = 0; k < symbols.Length; k++)

{

Console.WriteLine($"{count++,3}: {symbols[i]}{symbols[j]}{symbols[k]}");

}

}

}

Уравнение. Найти a и b.

for (double a = 1; a < 500; a++)

{

for (double b = 1; b < 500; b++)

{

if ((2 \* a \* a \* a + 3 \* b \* b) == 8337) Console.WriteLine($"a={a} b={b}");

}

}

Break и continue. Выполняются к ближайшему циклу. И помогает оптимизировать код, пропуская итерацию циклов.  
 int count = 0;

for (int i = 0; i < 10; i++)

{

if (i % 2 == 0) continue; //при условий

for (int j = 0; j < 10; j++)

{

Console.WriteLine(i + " " + j);

}

count++;

Console.WriteLine(count);

Двумерный массив = Матрица = Таблица.

int[,] matrix = new int[3, 4]; //создание двумерного массива. строки, столбцы.

matrix[1, 2] = 12; //присвоение 2 строка третий столбец

Random rnd = new Random();

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < 4; j++) {

//matrix[i,j] = rnd.Next(10);

Console.Write(matrix[i, j] + " ");

}

Console.WriteLine();

Создание строчек и столбцов пользователем.  
 Console.Write("Введите количество строк: ");

int row = int.Parse(Console.ReadLine());

Console.Write("Введите количество столбцов: ");

int col = int.Parse(Console.ReadLine());

int [,] arr2d = new int[row, col];

Random rand = new Random();

for (int i = 0; i < row; i++)

{

for (int j = 0; j < col; j++)

{

arr2d[i, j] = rand.Next(1,10);

Console.Write($"{arr2d[i, j]} ");

}

Console.WriteLine();

}

Массив из-за

string[,] table = new string[,] { //подставится автоматом

{ "00", "01", "02", "03" },

{ "10", "11", "12", "13" },

{ "20", "21", "22", "23" }

};

Console.WriteLine();

Console.WriteLine($"table.Rank = {table.Rank}"); //число измерений массива

Console.WriteLine($"table.GetLength(0) = {table.GetLength(0)}"); //количество строчек

Console.WriteLine($"table.GetLength(1) = {table.GetLength(1)}"); //количество столбцов

for (int i = 0; i < table.GetLength(0); i++) //от 0 по длине количества строчек

{

for (int j = 0; j < table.GetLength(1); j++) //от 0 по длине количества столбцов

{

Console.Write($"{table[i, j],3} "); //Тройка – отступить два символа

}

Console.WriteLine();

}

Элементы одномерного массива А из 9 чисел построчно расположить в матрице B [3x3]

int[] a = { 1, 2, 3, 4, 5, 6, 7, 8, 9 };

int[,] b = new int[3, 3];

int counterA= 0;

for (int i = 0; i < b.GetLength(0); i++)

{

for (int j = 0; j < b.GetLength(1); j++)

{

b[i, j] = a[counterA++];

Console.Write($"{b[i, j]} ");

}

Console.WriteLine();

}

Сумма элементов для каждой строки матрицы [4, 4]

Random random = new Random();

var A = new int[4, 4]; //создаем матрицу, 5 на 5

Console.WriteLine("исходная матрица А[4х4]: ");

for (int i = 0; i < A.GetLength(0); i++) //проходим по строчкам

{

int sum = 0; //создаем переменную для суммы строк

for (int j = 0; j < A.GetLength(1); j++) //проходим по столбцам

{

A[i, j] = random.Next(1, 5); //генерируем числа от 1 до 4

sum += A[i, j]; //складываем сумму в строке

Console.Write($"{A[i, j], 2}"); //выводим матрицу.

}

Console.WriteLine($" : {sum}"); // выводим сумму каждую строчку

}

Многомерные массивы. (кубик-рубик) .NET ВСЕГО 2 гб на массивы - Array.Length \* sizeof(double)

int[,,] matrix3D = new int[3, 4, 5];

Console.WriteLine($"matrix3D.Length = {matrix3D.Rank}");

Console.WriteLine($"matrix3D.GetLength(0) = {matrix3D.GetLength(0)}");

Console.WriteLine($"matrix3D.GetLength(1) = {matrix3D.GetLength(1)}");

Console.WriteLine($"matrix3D.GetLength(2) = {matrix3D.GetLength(2)}");

Console.WriteLine($"matrix3D.Length = {matrix3D.Length}"); //количество ячеек

Random r = new Random();

for (int i = 0; i < matrix3D.GetLength(0); i++) {

for (int j = 0; j < matrix3D.GetLength(1); j++) {

for(int k = 0; k < matrix3D.GetLength(2); k++)

{

matrix3D[i, j, k] = r.Next(1, 10);

Console.Write($"{matrix3D[i, j, k]} ");

}

Console.WriteLine();

}

Console.WriteLine();

}

Массив массивов (зубчатый массив), в каждой строке может содержать разной количество столбцов

int[][] jaggedArray = new int[3][]; //будет содержать в себе 3 массива

jaggedArray[0] = new int[2]; //в первой строке 2 элемента(столбца)

jaggedArray[0][0] = 1;

jaggedArray[0][1] = 3;

jaggedArray[1] = new int [4]; //во второй строке 4 элемента

jaggedArray[1][0] = 1;

jaggedArray [1][1] = 0;

jaggedArray[1][2] = 3;

jaggedArray[1][3] = 1;

jaggedArray[2] = new int[3]; //в третьей строке 3 элемента

jaggedArray[2][0] = 2;

jaggedArray[2][1] = 0;

jaggedArray[2][2] = 1;

for (int i = 0; i < jaggedArray.Length; i++)

{

for (int j = 0; j < jaggedArray[i].Length; j++)

{

Console.Write($"{jaggedArray[i][j]} ");

}

Console.WriteLine();

}

//программа для сбора и хранение информации о t воздуха

//в период с 9 до 21 ч вечера каждые 5 минут

//в период с 21 до 0 утра каждые 15 минут

Random r = new Random();

int[][] dataTime = new int[24][]; //создаем массив массивов каждого часа

for (int i = 0; i < dataTime.Length; i++)

{

dataTime[i] = (i >= 9 && i < 21) ? new int[12] : new int[4];

//если с 9 до 21, то 12 измерений в час, если нет то 4.

}

for (int j = 0; j < dataTime.Length; j++)

{

Console.Write($"{j}ч. ");

for (int k = 0; k < dataTime[j].Length; k++)

{

dataTime[j][k] = r.Next(20);

Console.Write($"{dataTime[j][k],3}");

}

Console.WriteLine();

}

foreach – помогает сократить код для вывода.   
недоцикл..нельзя внутри присвоить значения переменных.

Console.WriteLine("Cформировать маасив из N случайных чисел от -5 до 5");

Console.Write("Введите n:");

int n = int.Parse(Console.ReadLine());

Random r = new Random();

int[] numbs = new int[n];

for (int i = 0; i < n; i++)

{

numbs[i] = r.Next(-5, 6);

}

//массив сохранился со значениями

foreach (var e in numbs) //весь массив numbs

{

Console.Write($"{e} ");

}

foreach – помогает в массиве массивов

foreach (var array in dataTime) //есть массив - из массива массивов

{

foreach (var e in array) //элемент из текущего массива

{

Console.Write($"{e, 3} ");

}

Console.WriteLine();}

foreach string

for (int i = 0; i < message.Length; i++)

{

Console.Write($"{message[i]} ");

}

Console.WriteLine();

foreach (var c in message)

{

Console.Write($"{c} ");

}

Console.WriteLine();

Методы.

Отдельный кусочек кода выносится в метод, идет обращение к методу.  
Метод может быть создан внутри класса

УровниДоступа МодификаторыДоступа ВозврщаемыйТип ИмяМетода(ПараметрыМетода);

УровниДоступа: public, protected, internal, private;

МодификаторыДоступа: static, abstract, async, extern, override, virtual;

ВозвращаемыйТип: int, double, string и т.д.. void если ничего возвращать не нужно;

ИмяМетода: строчные и прописные буквы латинского алфавита, цифры, НО цифра не может быть на первом месте

ПараметрыМетода: аргументы, надо которыеми производятся каки-то действия.   
Формальные параметры – параметры указанные при описании метода. static void Print(string Text)

Фактические параметры – параметры указанные при вызове метода. Print("Фактические параметр");

private – по умолчанию

static void Main(string[] args)

{

Print("Фактические параметры");  
 }

Описательное имя для каждого слова с прописной буквы.

//Метод что-то делает

static void Delaet()

{

Console.ReadKey();

}

static public void HelloWorld()

{

Console.WriteLine("Hello world!");

}

//Метод что-то принимает аргументами, что-то делает

static void Print(string Text, int age)

{

Console.WriteLine($"Меня зовут {Text}, мне {age} лет");

}

//что-то делает, выдает результат своей работы

static string GetRandomString()

{

return Guid.NewGuid().ToString(); работы

}

static void Main(string[] args)

{

string randomText = GetRandomString();

Console.WriteLine(randomText);

}

/// <summary>

/// Что-то принимает, что-то делает, выдает результат своей работы.

/// </summary>

/// <param name="Count"></param>

/// <returns>случайную строку, ограниченную пользователем</returns>

static string GetRandomString(int Count)

{

return Guid.NewGuid().ToString().Substring(0, Count);

}

static void Main(string[] args)

{

Console.WriteLine(GetRandomString(14));

}

Сигнатура метода – возвращаемый тип, параметры метода.

Перегрузка методов – набор методов, имеющих одинаковое имя, отличающиеся количеством формальных параметров или их типом.

static void Fill(int[] Col)

{

Random r = new Random();

for (int i = 0; i < Col.Length; i++)

{

Col[i] = r.Next(100);

Console.Write(Col[i] + " ");

}

}

static void Fill(int[] Col, int Count)

{

Random r = new Random();

for (int i = 0; i < Count; i++)

{

Col[i] = r.Next(100);

Console.Write(Col[i]+" ");

}

}

static void Fill(int[] Col, int Count, string text)

{

Random r = new Random();

for (int i = 0; i < Count; i++)

{

Col[i] = r.Next(100);

Console.Write(text + " " + Col[i] + " ");

}

}

static void Main(string[] args)

{

int[] arr = new int[1000]; //количество чисел массива максимальное число

Fill(arr); //Fill(int[] Col)

Fill(arr, 300); //вывод массива с количеством символов 300, нельзя больше 999

Fill(arr, 200, "число массива"); //использует метод с 3-мя параметрами

}

Клавиша F12 поможет перебросить на нужный метод.

Params помогает в методе для массива не обозначать количество.  
  
 static int Sum(int Factor, params int[] Args) //int первым в методе, params в последний

{

int result = 0;

foreach (var e in Args)

{

result += e;

}

return result \* Factor;

}

static void Main(string[] args)

{

Console.WriteLine(Sum(2, 2, 3, 3, 4, 4, 53, 45, 34534));

}

Понятие рекурсивной функции – метода вызывающий сам себя.

Должен быть шаг рекурсии и базовый случай.

static void NotInfinity(int Count)

{

Console.Write("O\_- ");

Thread.Sleep(180);

Count = Count - 1;

if (Count > 0)

NotInfinity(Count);

}

static void Main(string[] args)

{

NotInfinity(100);

}

public static string Numbers(int i)

{

return i < 10 ? i.ToString() : $"{Numbers(i / 10)} {i % 10}";//хм...

}

static void Main(string[] args)

{

Console.WriteLine(Numbers(12312312));

}

­­

ДЗ

static String[] TextUserArray(string userText1)

{

return userText1.Split(' ');

}

static void TextSplitter(string userText2)

{

foreach (String s in TextUserArray(userText2))

{

Console.WriteLine(s);

}

}

static void Main(string[] args)

{

Console.WriteLine("Введите длинное предложение разделив слова - пробелом: ");

string userText = Console.ReadLine();

TextUserArray(userText);

TextSplitter(userText);

}

/// <summary>

/// Полученный текст разделяется на слова

/// </summary>

/// <returns></returns>

static String[] TextUserArray()

{

Console.WriteLine("Введите длинное предложение разделив слова - пробелом: ");

string userText = Console.ReadLine();

string[] arrayUserText = userText.Split(' ');

return arrayUserText;

}

/// <summary>

/// Инвертирование слов и их вывод

/// </summary>

/// <param name="array"></param>

static void TextReverse(Array array)

{

Array.Reverse(array);

foreach (var item in array)

{

Console.Write(item + " ");

}

}

static void Main(string[] args)

{

TextReverse(TextUserArray());

}

static string[] Split(string inputPhrase)

{

string[] arrayUserText = inputPhrase.Split(' ');

return arrayUserText;

}

static void ReversWords(string inputPhrase)

{

string[] array = Split(inputPhrase);

Array.Reverse(array);

foreach (var item in array)

{

Console.Write(item + " ");

}

}

static void Main(string[] args)

{

Console.WriteLine("Введите длинное предложение разделив слова - пробелом: ");

string inputPhrase = Console.ReadLine();

ReversWords(inputPhrase);

}

Встроенные методы. Класс Math и Convert, DateTime.

Convert преобразование из данных любого типа, в любой тип.

Класс Math.  
Предоставляет константы и статические методы для тригонометрических, логарифмических и иных общих математических функций.

Console.WriteLine(Math.PI); //число Пи

Console.WriteLine(Math.Min(1, 44)); //минимальное из чисел

Console.WriteLine(Math.Max(1, 858)); //максимальное из чисел

Console.WriteLine(Math.Floor(1.66)); //округление.меньше или равно указаному числу

Console.WriteLine(Math.Pow(22, 3)); //число вовзеденное в степень

Console.WriteLine(Math.Round(1.55543)); //округление до ближайшего целого

Console.WriteLine(Math.Round(22.323433443,3)); //сколько оставить знаков после запятой и округлить

DateTime. TimeSpan  
 DateTime date = new DateTime(2020, 04, 24, 17, 19, 22);

Console.WriteLine(date);

Console.WriteLine(date.Year); //вывод года на экран

Console.WriteLine(date.Month); //вывод месяца на экран

Console.WriteLine(date.Day); //вывод числа на экран

Console.WriteLine(date.Hour); //вывод часов на экран

Console.WriteLine(date.Minute); //вывод минут на экран

Console.WriteLine(date.ToShortTimeString()); //вывод времени на экран

Console.WriteLine(date.ToShortDateString()); //вывод даты на экран

DateTime newDate = date.AddMinutes(15); //добавить 15 минут

Console.WriteLine(newDate);

Console.WriteLine(DateTime.Now); //текущая дата и время компьютера

TimeSpan span = newDate - date;

Console.WriteLine(span);

Console.WriteLine(span.TotalMinutes);

date = DateTime.Now;

double sum = 0;

for (int i = 0; i <= 1000000; i++)

{

sum += i;

}

Console.WriteLine(sum);

TimeSpan timespan = DateTime.Now - date;

Console.WriteLine(timespan.TotalMilliseconds);

Работа со строками. Статический класс String, методы экземпляров.   
 string str = "Visual alusiV";

Console.WriteLine(str.IndexOf('i')); //поиск первого слева символа

Console.WriteLine(str.LastIndexOf('i')); //поиск последнего значения

Console.WriteLine(str.IndexOf('Z')); //-1 значит этого символа нет

string s = str.Insert(str.IndexOf(' '), " C#"); //после проблеа добавляет указанное

Console.WriteLine(s);

Console.WriteLine(str.Remove(6)); //удаляет все после 6 символа

Console.WriteLine(str.Remove(6, 3)); //удаляет 3 символа с 6 символа

Console.WriteLine(str.Substring(4)); //возвращает строку начиная с 4 символа

Console.WriteLine(str.ToUpper()); //все заглавные

Console.WriteLine(str.ToLower()); //все строчные

Console.WriteLine(str.Trim()); //удаляет пробелы в начале и конце

str.TrimEnd();

str.TrimStart();

string[] array = "1 2 3 4,5 6;7 8 9;10".Split(' ', ',', ';'); //разбивает на строки по указанным символам

for (int i = 0; i < array.Length; i++)Console.WriteLine(array[i]);

string test = null;

Console.WriteLine(string.IsNullOrEmpty(test)); //если она null или пустая -тру

test = string.Empty;

Console.WriteLine(string.IsNullOrWhiteSpace(test));

test = "Skill";

Console.WriteLine(string.IsNullOrEmpty(test));

char c = 'c';

Char.IsDigit(c); //десятичное число?

Char.IsLetter(c); //бука юникода?

Char.IsLower(c); //нижний регистр буква?

Char.IsUpper(c); //верхний регистр буква?

char[] symbols = "symbols".ToCharArray();

foreach (char item in symbols) Console.Write(item);

string stringg = new string(symbols); //создание строки из массива

Работа с файлами: File, fileinfo, directory, directoryinfo.  
 string text = File.ReadAllText(@"c:\adb\_run\data.txt"); //считывает данные

Console.WriteLine(text);

string[] lines = File.ReadAllLines(@"c:\adb\_run\data.txt"); //считывает построчно

foreach (var line in lines) Console.WriteLine(line);

lines = new string[] { "проверка 1", "проверка 2", "проверка 3" }; //записывает строки в файл

File.WriteAllLines(@"c:\adb\_run\data.txt", lines);

text = "проверка текста"; //записывает текста в файл

File.WriteAllText(@"c:\adb\_run\data.txt", text);

File.AppendAllLines; //добавляет строки в файл

File.AppendAllText; //добавляет текст в файл

File.Copy; //копирует

File.Delete; //удаляет

File.Exists; //проверяет существует ли?

File.Move; //перемещение

FileInfo – информация о файле.

FileInfo fileInfo = new FileInfo(@"C:\adb\_run\date.txt");

Console.WriteLine(fileInfo.FullName); //

Console.WriteLine(fileInfo.Name);

Console.WriteLine(fileInfo.DirectoryName);

Console.WriteLine(fileInfo.Attributes);

Console.WriteLine(fileInfo.Extension);

Console.WriteLine(fileInfo.IsReadOnly);

Console.WriteLine(fileInfo.LastAccessTime);

Console.WriteLine(fileInfo.LastWriteTime);

Directory

Directory.CreateDirectory;

Directory.Delete;

Directory.Exists;

Directory.Move;

static void GetDir(string path, string trim = "")

{

DirectoryInfo directoryInfo = new DirectoryInfo(path); //получаем инф о текущем каталоге

foreach (var item in directoryInfo.GetDirectories()) //перебираем все подкаталоги текущего каталога

{

Console.WriteLine($"{trim}{item.Name}"); //выводим информацию о нем

GetDir(item.FullName, trim + " "); //просмотр вложенного каталога

}

foreach (var item in directoryInfo.GetFiles()) //перебираем все файлы текущего каталога

{

Console.WriteLine($"{trim}{item.Name}"); //выводим информацию о них

}

}

static void Main(string[] args)

{

GetDir (@"C:\adb\_run\");

}