Var заделя една и съща памет за данни, докато като конкретизираме – заделяме по-малко спрямо типа, който ние определим

32 bits = 4 byte – 32 нули и единици

Ако имаме overflow – примерно при байта – след 255 се връща към 0 и продължава от там – от началото

Експлицитно конвертиране на данни – (type)

Няма смисъл да ползваме нещо по-малко от инт – има достатъчно памет. Само ако не достига числото ползваме по-големия тип long

Литерали – буква или поредица от символи преди число, което показват, че чилото е малко по-различен тип

0xFE – шеснайсетина бройна система

Ако напишем само 0 – числото е 0 по подразбиране integer. 0L – 0 long

Int number = 0; - ок

Int number = 0L – грешка!!! 0L е 64 битово число и няма как да го вкара в 32 бита клетка

0U – unsigned long – само положително

Floating point – плаваща запетая. Голямо или малко число с някаква точност (precision). Важно е колко значещи символа имаме след запетаята. Няма значение колко нули има – те не са значещи символа, а 10 на степен x

Float – 32 bit precision of 7 digits 0.0F

Double – 64 bit precision of 15-16 digits 0.0D – но това е default тип данни и можем да го изпуснем

По същия начин като int – тук ползваме основно double.

Math – библиотека с неща от математиката.

Math.Round – round to integer mathematically!

Math.Ceiling – закръгля нагоре до следващо значещо цяло число

Math.Floor – надолу

B = 1.23456789 Math.Floor ( b \* 1000 ) / 1000 - ….. = 1.2345

0:f12 – floating point 12 символа

Int nuber = int.MaxValue;

Int number = int.MinValue;

Scientific notation – 1e+34 = 1 \*10^34

10 / 4 – интегрално, целочислено деление

Едно от двете числа ако е реално става реално делението

Цяло число делено на нула – грешка

Реално число делено на нула – безкрайност

0.0 / 0.0 = NaN;

NaN – not a number

Decimal – 0.0M – 128 bit, precision of 28-29 digits

Type conversion – implicit – automatic and lossless, and explicit - lossy (type)

Int courses = (int) Math.Ceiling((double)n / p );

Bool – False = 0; True = 1;

Булевите променливи не могат да се събират – те са ЛОГИЧЕСКИ, а не математически израз

Char – takes 16 bits of memory – 4 числа в шеснайстична бройна система – от U+0000 до U+FFFF. Unicode character. ASCII е част от UTF и отговаря за латинската част от таблицата.

UTF – стандарт по който изобразяваме всеки един символ. Включва ASCII + килилица +японски+китайски+.......

(int) char = номера на символа в таблицата

Escaping characters

\t = TAB

\n = new line

\” – ескейпва кавичка...

\ - покажи символа такъв, какъвто е, не използвай предназначението му

\uXXXX – показва уникод символа

\u8449 – символа листо на китайски

String – низ от символи, масив от символи, може да се итерира като такъв.

3 начина на изобразяване

“” – всичко вътре е стринг

Verbatim – дословно - @......... – ескейпваме само кавичката с втора кавичка

Interpolated strings – вътре в стронговета вкарваме променливи с долар отпред – $”{firstName} {lastName}”;

Наименоване – camelCase се нарича конвенцията камила с гърбица – първо мака буква, после голяма

СЪществително или прилагателно + още едно съъществително

Булевите променливи обикновена са – isSomething. Примерно – вместо loggedUser - isLogged.

Живота на променливите е между 2 къдрави скоби!!!

Span – обхват на променливата, времето, в които тези промениви въжат. Трябва да ги дефинираме ВЪЗМОЖНО НАЙ-КЪСНО!!! В момента, в който ни трябват и не по-рано! Примерно span от 6 реда може да е съкратен на спан от 2 реда.

Въпроси:

(int) е по-скоро за конверсия между числа. Int.Parse е конвертиране на стринг към число

{0:f2} – визуализира 2 символа. Math.Round – променя изцяло стойността, която се държи в клетката.

Виж видеото с частта за наименоване!!! И началото!!!