Типове данни и променливи

Реални типове

Учителски екип

Обучение за ИТ кариера

https://it-kariera.mon.bg/e-learning/

https://github.com/BG-IT-Edu/School-Programming/tree/main/Courses/Applied-Programmer/Programming-Fundamentals

Съдържание

1. Реални типове с плаваща запетая
2. Аномалии при изчисления с плаваща запетая
3. Реален тип с десетична точност

double

float

decimal

Реални числени типове

И как да ги ползваме в C#

Какво са типовете с плаваща запетая?

double

float

- Типовете с плаваща запетая:
  - Съдържат реални числа, например: `1.25`, `-0.38`
  - Имат диапазон и точност според използваната памет
  - Понякога се наблюдават аномалии при изчисления
  - Могат да пазят много малки и много големи стойности като `0.00000000000001` и `10000000000000000000000000000.0`

Числа с плаваща запетая

- Типовете с плаваща запетая са:
  - **`float`** ($\pm 1.5 \times 10^{-45}$ до $\pm 3.4 \times 10^{38}$)
    - 32-битов, точност 7 знака след запетаята
  - **`double`** ($\pm 5.0 \times 10^{-324}$ до $\pm 1.7 \times 10^{308}$)
    - 64-бита, точност от 15-16 знака след запетаята
- Стойността по подразбиране е:
  - **`0.0F`** за тип **`float`**
  - **`0.0D`** за тип **`double`**

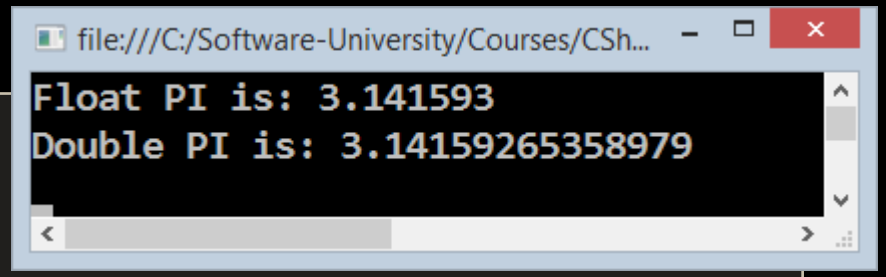
double

float

Точност на Пи – Пример

- Разликата в точността, когато ползваме **float** и **double**:

```
float floatPI = 3.14159265358979238f;
double doublePI = 3.14159265358979238;
Console.WriteLine("Float PI is: {0}", floatPI);
Console.WriteLine("Double PI is: {0}", doublePI);
```



- Забележете наставката „**f**“ след числото на първия ред!
  - Реалните числа по подразбираме се възприемат за **double**!
  - Ако желаем дадена стойност да се запише като **float**, трябва изрично да я преобразуваме

Закръгляне на числа с плаваща запетая

- `Math.Round(3.45)` – закръгля към цяло число(математически)
- `Math.Round(2.3455, 3)` – закръгляне с точност
- `Math.Ceiling()` – закръгля нагоре към най-близкото цяло число
- `Math.Floor()` – закръгля надолу към най-близкото цяло число

2.5 → 3
3.5 → 4
3.45 → 3

```
double a = 2.3455;
Console.WriteLine(Math.Round(a));    // result: 2
Console.WriteLine(Math.Round(a, 3)); // result: 2.346
Console.WriteLine(Math.Ceiling(a));  // result: 3
Console.WriteLine(Math.Floor(a));    // result: 2
```

Задача: Лице на кръг (с точност 12 знака)

- Напишете програма, в която да въведете радиус **r** (реално число) и изведете лицето на кръга с точност 12 знака след запетаята:

2.5	→	19.634954084936

1.2	→	4.523893421169

- Примерно решение:

```
double r = double.Parse(Console.ReadLine());
Console.WriteLine("{0:f12}", Math.PI * r * r);
```

Тествайте в Judge: https://judge.softuni.bg/Contests/2648

Експоненциален запис

- Числата с плаваща запетая могат да ползват експоненциален запис, например:
  1e+34, **1E34**, **20e-3**, **1e-12**, **-6.02e28**

```
double d = 10000000000000000000000000000000000.0;
Console.WriteLine(d); // 1E+34

double d2 = 20e-3;
Console.WriteLine(d2); // 0.02

double d3 = double.MaxValue;
Console.WriteLine(d3); // 1.7976931348623E+308
```

Делене с плаваща запетая

- Целочисленото деление и делението на числа с плаваща запетая са различни операции

```
Console.WriteLine(10 / 4);     // 2 (целочислено)
Console.WriteLine(10 / 4.0);   // 2.5 (реално)
Console.WriteLine(10 / 0.0);   // Infinity
Console.WriteLine(-10 / 0.0); // -Infinity
Console.WriteLine(0 / 0.0);    // NaN (не е число)
Console.WriteLine(8 % 2.5);    // 0.5 (3 * 2.5 + 0.5 = 8)

int d = 0;                     // Целочисленото деление
работи по друг начин!
Console.WriteLine(10 / d); // DivideByZeroException
```

Аномалии при изчисления с плаваща запетая

- Понякога изчисленията работят неправилно!

```
Console.WriteLine(100000000000000.0 + 0.3);
// Result: 100000000000000 (загуба на точност)

double a = 1.0f, b = 0.33f, sum = 1.33;
Console.WriteLine("a+b={0} sum={1} equal={2}",
  a+b, sum, (a+b == sum));
// a+b=1.33000001311302 sum=1.33 equal=False

double one = 0;
for (int i = 0; i < 10000; i++) one += 0.0001;
Console.WriteLine(one); // 0.999999999999906
```

Реален тип с десетична точност

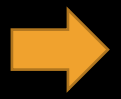
- Има специален реален тип с десетична точност в C#:
  - **`decimal`** ($\pm 1{,}0 \times 10^{-28}$ до $\pm 7{,}9 \times 10^{28}$)
    - 128-битов, с точност до 28-29 знака
  - Използва се за финансови изчисления
  - Почти няма грешки при закръгляне
  - Почти няма загуба на точност
- Стойността по подразбиране за **`decimal`** е:
  - **`0.0M`** (**`M`** е наставката за десетичните числа)

decimal

Задача: Точна сума на реални числа

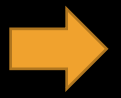
- Напишете програма, която да въвежда **n** числа и да изведете тяхната точна сума:

```
2
100000000000000000000
5
```



```
100000000000000000005
```

```
2
0.00000000003
333333333333.3
```



```
333333333333.30000000003
```

Тествайте в Judge: https://judge.softuni.bg/Contests/2648

Решение: Точна сума на реални числа

- Този код работи, но понякога прави грешки при закръгляне:

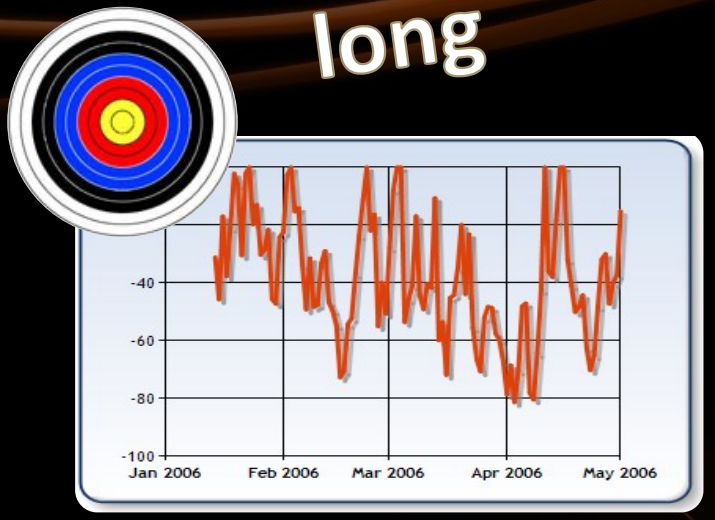
```
int n = int.Parse(Console.ReadLine());
double sum = 0;
for (int i = 0; i < n; i++)
  sum += double.Parse(Console.ReadLine());
Console.WriteLine(sum);
```

- Сменете **double** с **decimal** и вижте разликите

double

int

long



float

decimal

Цели и реални числа

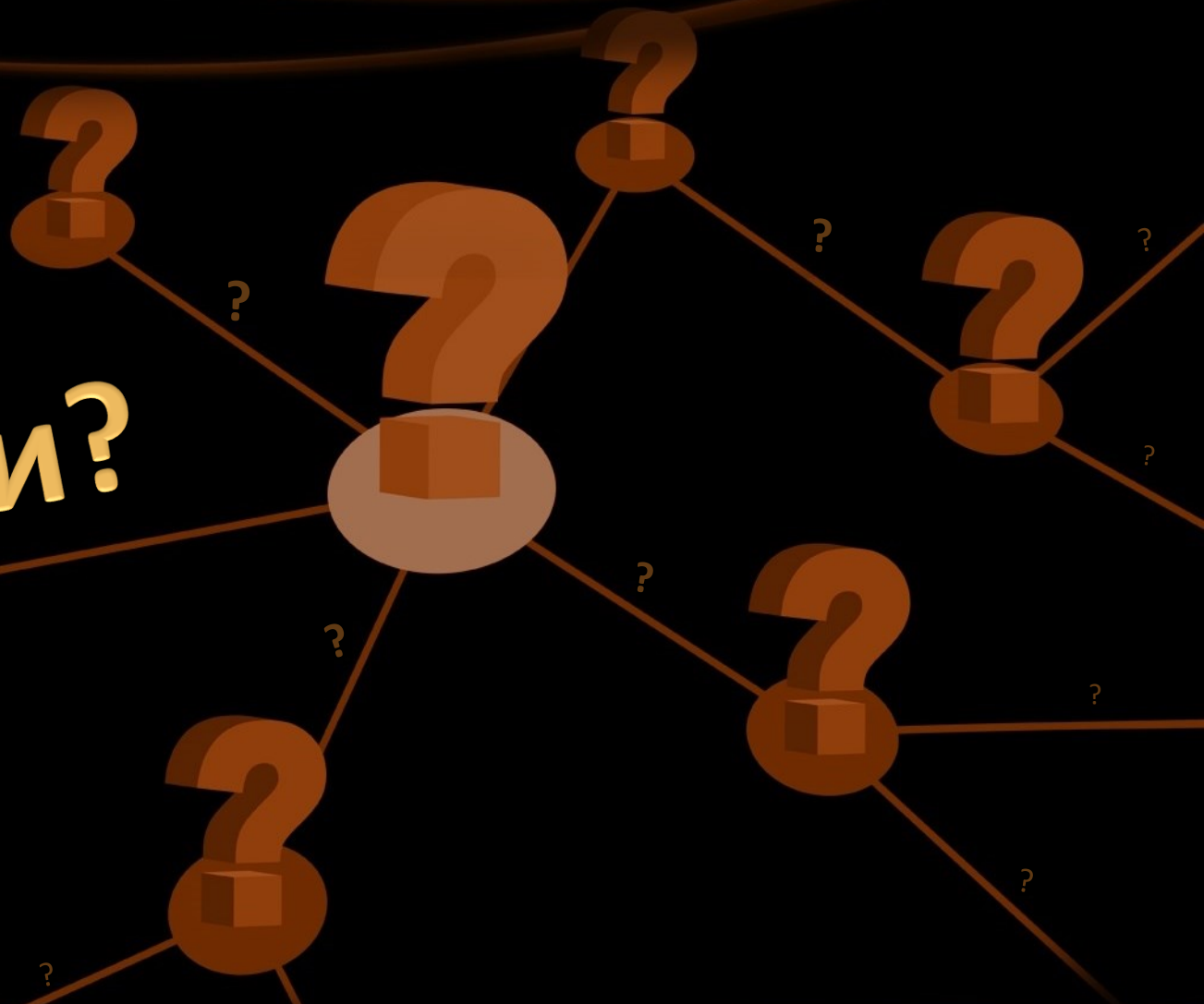
Експерименти

Какво научихме днес?

- Класически типове данни:
  - Реални типове с плаваща запетая: съдържат реални числа
    - Имат определена точност
    - Може да се наблюдават аномалии
  - Реален тип с десетична точност: съдържат реални числа
    - Има по-висока точност
    - Много по-малко вероятно да се наблюдава аномалия или загуба на точност

Бройни системи

Въпроси?



https://github.com/BG-IT-Edu/School-Programming/tree/main/Courses/Applied-Programmer/Programming-Fundamentals

Министерство на образованието и науката (МОН)

- Настоящият курс (презентации, примери, задачи, упражнения и др.) е разработен за нуждите на Национална програма "**Обучение за ИТ кариера**" на МОН за подготовка по професия "Приложен програмист"

- Курсът е базиран на учебно съдържание и методика, предоставени от **фондация "Софтуерен университет"** и се разпространява под свободен лиценз **CC-BY-NC-SA**