Статични елементи на класа
(полета, свойства, методи, конструктори)

Съдържание

1. Статични полета
2. Статични свойства
3. Статични методи
4. Статични конструктори

Задача: Преброй хората

- Напишете програма, която да поддържа информация колко обекта от клас Person има създадени до момента.
- Подсказка: Ще използваме статично поле и свойство.

Статични полета

- Принадлежат на самия клас
- Имат една и съща стойност за всеки обект от класа
- Могат да бъдат достъпени и само чрез класа - без създаване на обект от този клас

Статични свойства

- Принадлежат на самия клас
- Могат да бъдат достъпени и само чрез класа - без създаване на обект от този клас

Статични свойства

Използването на свойства е удобно, когато имаме статични полета, но не искаме да позволим тяхната промяна от друг клас. Ако трябва да използваме само поле, то за да го достъпим извън класа, полето трябва да е public, което пък ще позволи неговото изменение. Затова ползваме статичните свойства.

Решение: Преброй хората (1)

1. Ще създадем статично поле, което ще пази информацията.
2. След това ще направим статично свойство, което ще има само get, понеже в противен случай ще можем да манипулираме брояча, когато използваме класа, а в случая идеята е ползвателя на класа да не може да промени полето, в което е записан броя, а само да го достъпи.
3. Броячът ще се увеличава в конструктора на класа.

Решение: Преброй хората (2)

```
class Person {
  // останалите части на класа са пропуснати
  private static int count = 0;
  public Person(string name, int age) {
    // в конструктора променяме стойността на count
    Person.count += 1;
  }
  public static int Count { // статично свойство
    get { return count; }
  }
}
```

Задача: Аритметични действия

Създайте клас, който поддържа методи за аритметични действия върху две цели числа:

- Add(int, int) – събира числата, подадени като параметри
- Multiply(int, int) – умножава числата.

Използвайте методите от този клас в Main метода, за да въведете команда, числата и да извършете операцията.

Подсказка: Тъй като не се пази никаква информация, няма смисъл да се създават обекти. Ще използваме статични методи.

Статични методи

- Принадлежат на самия клас
- Могат да бъдат достъпени само чрез класа - без създаване на обект от този клас
- Удобни са за извършване на действия върху всички обекти от класа или за извършване на действия, които нямат пряко отношение към обектите

Решение: Аритметични действия

```
class Arithmetics {
  public static int Add(int a, int b){
    return a+b;
  }

  public static int Multiply(int a, int b) {
    return a * b;
  }
}
```

- Извиквайте методите по аналогичен начин в `Main()`:

```
var result = Arithmetics.Add(10, 15);
```

Статични класове

- В решението оставихме класа нестатичен. Това означава, че от него може да се създаде обект. В случая това е безсмислено.
- За да не се допуска създаване на обект от клас, който има само статични елементи, можем да го направим статичен: static class.
- Когато отбележим един клас като статичен, това означава, че неговите елементи също ще са статични и от този клас няма да може да се създават обекти, а членовете ми ще може да се ползват само статично. Много класове от .NET са статични (например Math)

Статични конструктори

- Конструкторите в един клас също могат да бъдат статични
- Ако един конструктор е статичен, той се изпълнява, когато за първи път се използва класа. Например:
  - Създаде се обект от класа (ако класа е нестатичен)
  - Достъпва се статичен член от класа

  Най-често статичните конструктори се използват за инициализация на статични полета

Задача: Заявка за корен

Напишете клас, който съдържа метод, който връща корен квадратен при подадена заявка. Възможно е да получите голям брой заявки, така че трябва да отговаряте бързо на всяка една от тях.

- Подсказка: За не губим време при заявките, предварително ще извършим времеемките изчисления в нещо като кеш памет. Така ще можем да насмогнем на повишения брой заявки.

Решение: Заявка за корен

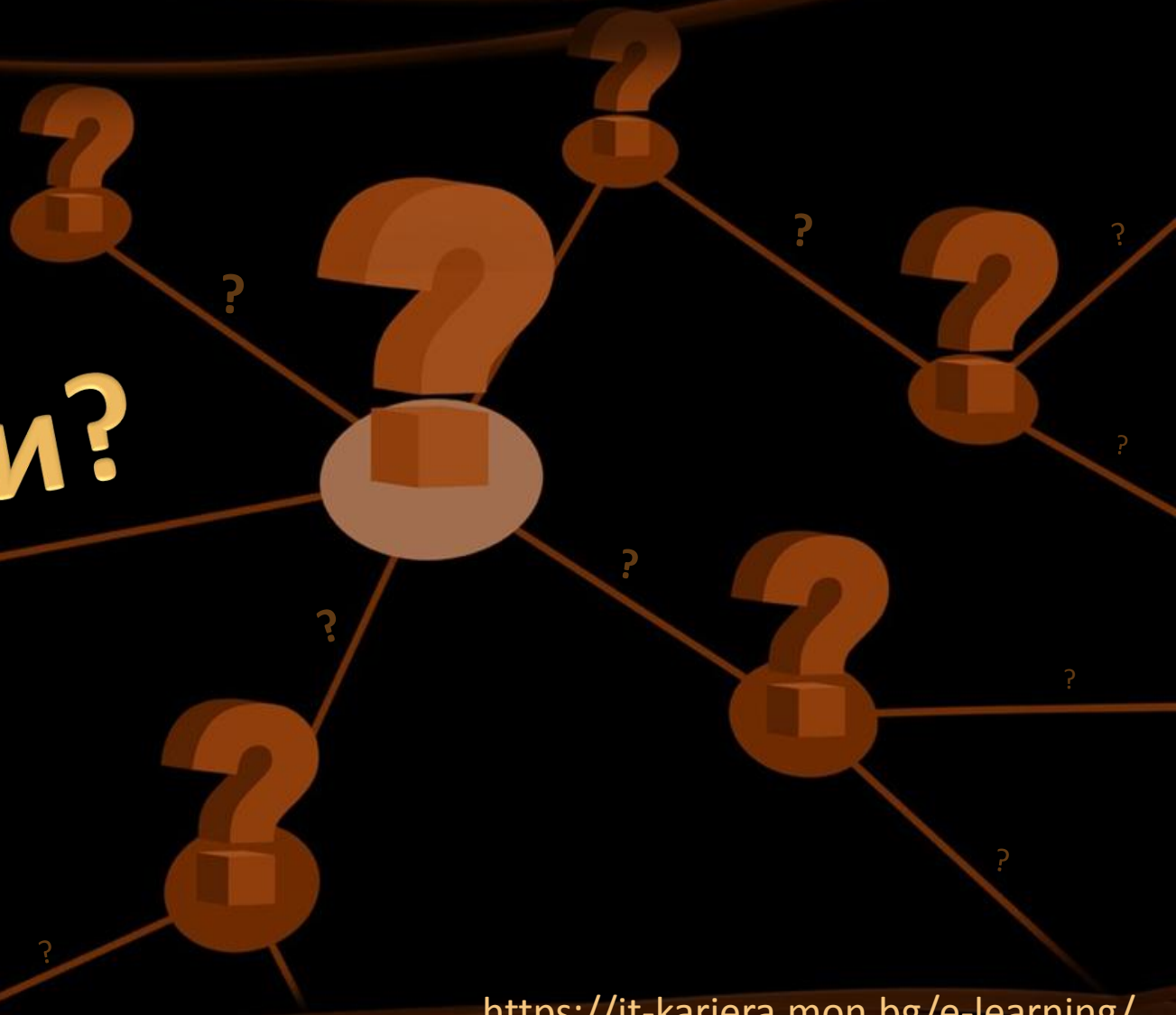
```
public static class SquareRootPrecalculator {
  public const int MaxValue = 1000;
  private static double[] sqrtValues;

  static SquareRootPrecalculator() {
    sqrtValues = new double[MaxValue+1];
    for (int i = 1; i <= MaxValue; i++)
      sqrtValues[i] = Math.Sqrt(i);
  }
  public static double GetSqrt(int value) {
    return sqrtValues[value];
  }
}
```

Какво научихме?

- Статични полета:
  - Еднаква стойност за всички обекти от класа. Достъп чрез самия клас.
- Статичните свойства:
  - Принадлежат на класа. Достъп до данни от статични полета.
- Статичните методи:
  - Принадлежат на класа. Достъп - чрез името на класа.
- От статичен клас не може да се създаде обект.
- Статичните конструктори се изпълняват, когато за първи път се създаде обект от класа или се достъпи негов статичен член

Статични елементи на класа

Въпроси?



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

Лиценз

- Настоящият курс (слайдове, примери, видео, задачи и др.) се разпространяват под свободен лиценз "Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International"

- Благодарности: настоящият материал може да съдържа части от следните източници
  - Книга "Основи на програмирането със C#" от Светлин Наков и колектив с лиценз CC-BY-SA