

# T03. Pagrindiniai C# elementai. Klasės, metodai, objektai.

4 ak. val.

# Temos klausimai

1. Baziniai duomenų tipai. Operacijos ir reiškiniai.
2. Loginiai reiškiniai.
3. Sąlygos operatorius.
4. Ciklo operatoriai.
5. Duomenų įvestis ir išvestis.
6. Klasės sudarymo taisyklės.
7. Metodų sudarymo taisyklės.
8. Sąsajos metodai.
9. Programavimo stilius, kultūra.



# *Baziniai duomenų tipai. Operacijos ir reiškiniai*

# Baziniai duomenų tipai (1/2)

Tipas	Atminties kiekis	Pastabos
<code>int</code>	32 bitai	<code>int.MinValue</code> , <code>int.MaxValue</code>
<code>double</code>	64 bitai	<code>double.MinValue</code> , <code>double.MaxValue</code>
<code>decimal</code>	128 bitai	<code>decimal.MinValue</code> , <code>decimal.MaxValue</code> konstantos <code>sk3 = 45m</code> ; arba <code>sk3 = 45M</code> ;
<code>char</code>	16 bitų	Unicode – Windows, Visual Studio – UTF-16
<code>string</code>		Eilutė
<code>bool</code>	8 bitai	<code>true</code> ; <code>false</code>

# Baziniai duomenų tipai (2/2)

Tipas	Numatyta reikšmė	Min reikšmė	Max reikšmė
<code>int</code>	0	-2147483648	2147483647
<code>double</code>	0.0d	$\pm 5.0 \times 10^{-324}$	$\pm 1.7 \times 10^{308}$
<code>decimal</code>	0.0m	$\pm 1.0 \times 10^{-28}$	$\pm 7.9 \times 10^{28}$
<code>char</code>	'\u0000'	'\u0000'	'\uffff'
<code>string</code>	null	-----	-----
<code>bool</code>	false	false; true	

# Operacijos

Operacijos leidžia atlikti veiksmus su operandais.

Operandai – tai kintamieji ir/arba konstantos (*galimi ir kiti atvejai, apie tai sužinosime vėliau*).

Veiksmai atliekami tarp dviejų suderinamo tipo operandų. Operacijos rezultato tipas yra suderinamas su operandų tipais.

Sveiko tipo konstanta gali būti ir realiu skaičiumi.

# Aritmetinių operacijų žymėjimas

Operacija	Operacijos pavadinimas
+	sudėtis
-	atimtis
*	daugyba
/	dalyba. Dviejų sveikųjų dydžių rezultatas – sveikasis, apvalinimo nėra
%	sveikųjų skaičių dalybos liekana. Galima ir su realiaisiais skaičiais
Koks 11.12 % 3 rezultatas?	

# Prioritetai

Operacija	Pavadinimas	Pastabos
Atliekamos pirmiau		Jei yra kelios šio tipo operacijos, atliekamos iš kairės į dešinę
*	daugyba	
/	dalyba	
%	liekana	
Atliekamos vėliau		
+	sudėtis	Jei yra kelios šio tipo operacijos, atliekamos iš kairės į dešinę
-	atimtis	



# Reiškiniai

Reiškinį gauname operacijomis apjungę operandus.

Operacijos reiškinyje atliekamos tam tikra tvarka pagal jų prioritetą.

Kai nustatyti operacijų atlikimo tvarkai prioriteto nepakanka arba operacijos turi būti atliekamos kita tvarka, nei nustato prioritetai, naudojami skliaustai. Jų gylis neribotas.

# Reiškinių pavyzdžiai

**int** obuolys, kriaušė, k, p;

**double** x, y, z, a = 11.12, b = 17.88;

k / p // Atsakymas – sveikas skaičius

k / (p \* 1.0) // Atsakymas – realus

obuolys + kriaušė

x + y/z

(x + y)/z

a\*a + 2\*b\*x - a\*b\*x

a % 4 + b      Koks atsakymas?

# Priskyrimo operacija

Apskaičiavus reiškinį, gautą reikšmę reikia įsiminti. Tam naudojama priskyrimo operacija:

```
int obuolys, kriaušė, kompotas;
```

```
double x, y, z, a, b, t, g;
```

```
kompotas = obuolys + kriaušė;
```

```
t = x + y/z;
```

```
g = (x + y)/z;
```

```
z = a*a + 2*b*x - a*b*x;
```

```
g = t = b = (x + y)/z;
```

# Prieaugio ir mažėjimo operatoriai

Operacija	Pavadinimas	Pavyzdys	Paaiškinimas
<b>++</b>	Prefiksinis prieaugis	<code>++x</code>	Didina reikšmę 1, o po to reiškinyje naudoja naują
<b>++</b>	Pofiksinis prieaugis	<code>x++</code>	Naudoja reiškinyje esamą reikšmę, o po to didina 1
<b>--</b>	Prefiksinis mažėjimas	<code>--x</code>	Mažina reikšmę 1, o po to naudoja reiškinyje naują
<b>--</b>	Pofiksinis mažėjimas	<code>x--</code>	Naudoja reiškinyje esamą reikšmę, o po to mažina 1
<b>Tinka visiems skaitmeniniams tipams</b>			



## *Loginiai reiškiniai*

# Sąlyga

Tai reiškiny, kuriame dalyvauja aritmetinės ir palyginimo operacijos, ir kurio reikšmė įgauna vieną iš dviejų galimų reikšmių: **true** arba **false**

Pvz.,  **$i < 10$** ,  **$a + b < c$** ,  **$a == 1$**

Gali būti naudojamos loginės operacijos ( **$\&\&$** ,  **$||$** ,  **$!$** ).

Tokie reiškiniai naudojami priskyrimo ( **$=$** ), ciklo (**for**, **while**) ir sąlygos tikrinimo (**if**) sakiniuose.

# Sąlygos operacijos

Ženklas	Operacijos pavadinimas
<b>==</b>	ar lygu
<b>!=</b>	ar nelygu
<b>&gt;</b>	ar daugiau
<b>&lt;</b>	ar mažiau
<b>&gt;=</b>	ar daugiau arba lygu
<b>&lt;=</b>	ar mažiau arba lygu

# Loginis kintamasis

Loginis kintamasis aprašomas kaip  
**bool vardas;**

Jo galimos reikšmės yra **true** arba **false**

Pavyzdys:

```
bool a = true, b = false;
```



# Loginės operacijos

- Sąlyginė (Conditional) daugyba (ir):  
 $\&\&$
- Sąlyginė sudėtis (arba):  
 $\|\|$
- Loginis neigimas (ne):  
 $!$
- Loginė (boolean logical) daugyba (ir):  
 $\&$
- Loginė sudėtis (arba):  
 $|$
- Suma modulių 2 (Xor):  
 $\wedge$

# Loginės operacijos ! lentelė

X	!X
false	true
true	false

# Loginės operacijos && lentelė

X1	X2	X1 && X2
false	false	false
false	true	false
true	false	false
true	true	true

# Loginės operacijos || lentelė

X1	X2	X1    X2
false	false	false
false	true	true
true	false	true
true	true	true

# Loginė operacija $\wedge$

X1	X2	$X1 \wedge X2$
false	false	false
false	true	true
true	false	true
true	true	false

# Loginiai reiškiniai (1/2)

Loginių operacijų  $\&$ ,  $|$  lentelės yra tokios pat, kaip ir atitinkamų operacijų  $\&\&$ ,  $||$ . Jas nagrinėsite vėliau, skaitmeninės logikos paskaitoje.

Loginiai reiškiniai gaunami loginius operandus jungiant loginių operacijų ženklais. Veiksmų atlikimo tvarka reguliuojama operacijų prioritetais ir skliaustais.

Palyginimo ir loginių veiksmų prioritetai:

1.  $<$ ,  $>$ ,  $<=$ ,  $>=$
2.  $==$ ,  $!=$
3.  $\&\&$ ,  $\wedge$ ,  $||$

# Loginiai reiškiniai (2/2)

Loginiai operandai, tai:

- loginės reikšmės
- loginiai kintamieji
- loginiai reiškiniai skliaustuose

Pavyzdys:

```
int a = 10, b = 1, x;  
bool c = a > 0 && (b < -10 || b > 0);  
x = (2 * a + b) % 5;  
if(x >= 0 && x <= 3 || c)  
    Console.WriteLine(" c={0} x={1}", c, x);
```

# Loginių reiškinių naudojimas

Sąlygos tikrinimo sakiniai:

```
int a = 10, b = 1;  
if (a > 0) Console.WriteLine("Daugiau");
```

Ciklo sakiniai (ciklo kitimo sąlyga):

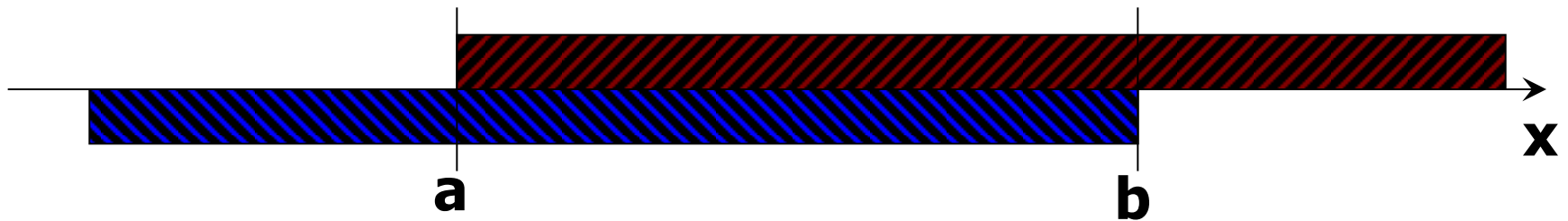
```
int a = 10, b = 1;  
for (int i = 0; i <= b; i++)  
    Console.WriteLine ("***");  
while (b < a--)  
    Console.WriteLine("????");
```



# Populiarūs loginiai reiškiniai (1)

$x \in [a; b]$  (priklauso intervalui):

**$a \leq x \ \&\& \ x \leq b$**



# Populiarūs loginiai reiškiniai (2)

$x \notin [a; b]$  (nepriklauso intervalui):

$$x < a \mid \mid x > b$$



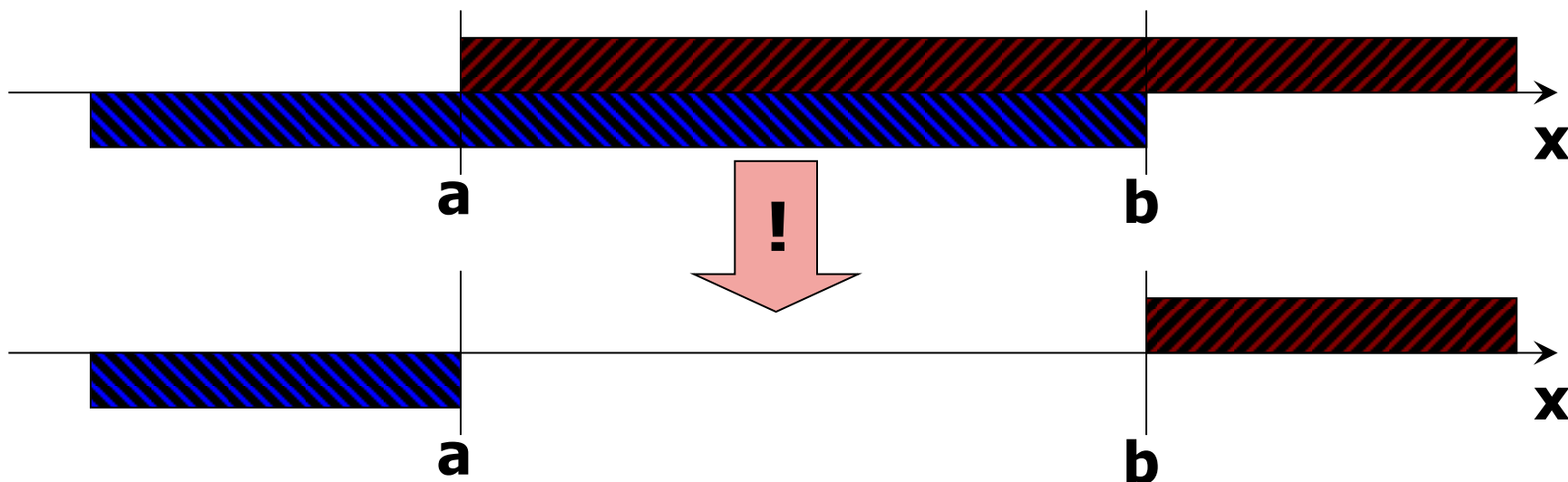
# Populiarūs loginiai reiškiniai (3)

Kadangi loginis neigimas duoda priešingą reikšmę, tai

$$x < a \ || \ x > b$$

yra tapatu

$$\neg(a \leq x \ \&\& \ x \leq b)$$



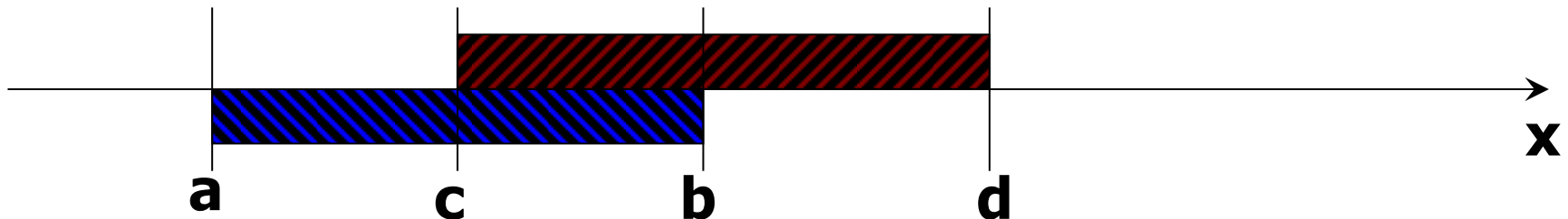
# Populiarūs loginiai reiškiniai (4)

x priklauso intervalų sankirtai, t.y.

$$x \in [a; b] \text{ ir } x \in [c; d]:$$

**$(a \leq x \ \&\& \ x \leq b) \ \&\&$**

**$(c \leq x \ \&\& \ x \leq d)$**



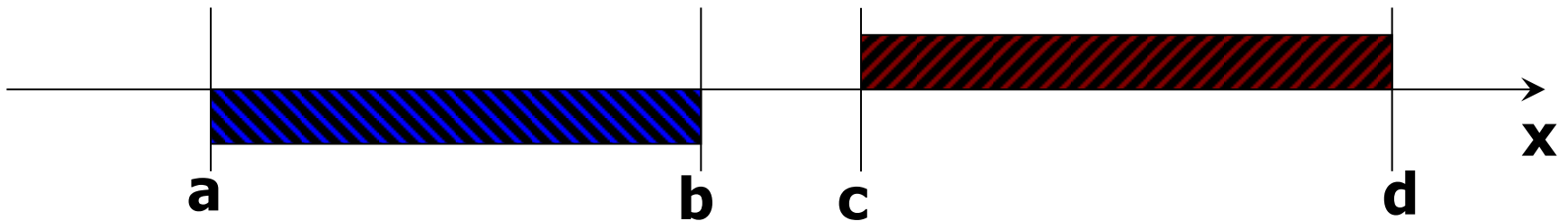
# Populiarūs loginiai reiškiniai (5)

$x$  priklauso vienam iš intervalų, t.y.

$x \in [a; b]$  arba  $x \in [c; d]$ :

`(a <= x && x <= b) ||`

`(c <= x && x <= d)`



# Loginio reiškinių rezultatas

Loginio reiškinių rezultatas gali būti ne tik tiesiogiai naudojamas sąlygos tikrinimo ar ciklo sakiniuose, bet ir priskiriamas loginiam kintamajam:

```
bool yra;
```

```
yra = (a <= x && x <= b) ||  
      (c <= x && x <= d);
```

# Operacijų prioritetai

Tipas	Operacijos ženklas	Prioritetas
Vienanarės Aritmetinės	!, ++, --, +, - *, /, % +, -	Aukščiausias
Palyginimo	<, >, <=, >= ==, !=	
Loginės	&& 	
Priskyrimo	=, +=, -=, *=, /=, %=	
		Žemiausias



# *Sąlygos operatorius if*



# Sąlygos operatorius **if**

**if** (Sąlyga)

**PirmasSakinys;**

**else**

**AntrasSakinys;**

Jei pirmo ar antro sakinio vietoje turi būti keletas sakinių, jie skliaudžiami skliaustais { }

---

**if** (Sąlyga)

**PirmasSakinys;**

# Sąlygos operatorius **if**

```
if (Sąlyga)  
    ;  
else  
    AntrasSakinys;
```

---

```
if (Sąlyga)  
    { }  
else  
    AntrasSakinys;
```

# Operatoriaus **if** pavyzdys (1)

```
if (a <= 0)
    ;
else {
    S = S + a;
    k++;
}
```

```
if (a <= 0)
    { }
else {
    S = S + a;
    k++;
}
```

Įspėjimas: Galimai klaidinga tuščia išraiška

## Operatoriaus **if** pavyzdys (2)

Patikrinkite, ar taškas (x, y) yra apskritime, apibrėžtame spinduliu r aplink koordinatinių centrą.

```
if (x*x + y*y <= r*r)
    Console.WriteLine ("Taškas yra apskritime");
else
    Console.WriteLine ("Taškas ne apskritime");
```

# Operatoriaus **if** pavyzdys (3)

Duotos dviejų taškų  $(x_1, y_1)$  ir  $(x_2, y_2)$ , esančių pirmame ketvirtyje, koordinatės. Kuris taškas yra daugiau nutolęs nuo x ašies?

```
if (y1 > y2)  
    Console.WriteLine ("Pirmas taškas");  
else  
    Console.WriteLine ("Antras taškas");
```

# Operatoriaus **if** pavyzdys (4)

Apskaičiuoti kintamąjį y:

$$y = \begin{cases} \sin x, & x < -1 \\ \cos x, & -1 < x < 3 \\ \sin(x+1)^2, & 3 \leq x < 10 \\ \cos 2x, & \text{likusiais atvejais} \end{cases}$$

```
if (x < -1)
    y = Math.Sin(x);
else if (-1 < x && x < 3)
    y = Math.Cos(x);
else if (3 <= x && x < 10)
    y = Math.Pow(Math.Sin(x + 1), 2);
else
    y = Math.Cos(2 * x);
Console.WriteLine ("x = {0,8:f2}    y = {1, 8:f4}", x, y);
```

# Matematinės funkcijos

Metodas	Pavyzdys
Abs(x)	Abs(24.5) yra 24.5, Abs(-24.5) yra 24.5
Ceiling(x)	Ceiling (9.2) yra 10, Ceiling (-9.2) yra -10.0
Floor(x)	Floor (9.2) yra 9, Ceiling (-9.2) yra -9.0
Sin(x)	Sin(0.0) yra 0.0 – argumentas radianais
Cos(x)	Cos(0.0) yra 1.0 – argumentas radianais
Tan(x)	Tan(0.0) yra 0.0 – argumentas radianais
Exp(x)	Exp(1.0) yra 2.71828
Log(x)	Log(Mat.E) yra 1.0
Max(x, y)	Max(2.4, 15.6) yra 15.6
Min(x, y)	Min(2.4, 15.6) yra 2.4
Pow(x, y)	Pow(2.0, 6.0) yra 64.0, Pow(9.0, 0.5) yra 3.0
Sqrt(x)	Sqrt(9.0) yra 3.0

# Matematinų funkcijų klasė

Matematinų funkcijų klasė yra **Math**.

Kubinės šaknies traukimas

```
sk2 = Math.Pow(sk2, 1.0/3); //Kubinė šaknis
```

```
sk3 = Math.Pow(sk3, 0.25); //4 laipsnio šaknis
```

Atsakymai: sk2, sk3 – realaus tipo.

```
Math.PI – 3.14159265358979
```

```
Math.E – 2.71828182845905
```



# Savarankiško darbo užduotis

- Patikrinkite, ar galima iš **x** ištraukti kvadratinę šaknį?
- Patikrinkite, ar realūs skaičiai **a** ir **b** lygūs?
- Patikrinkite, kiek sprendinių turi lygtis  
 **$a * x^2 + b * x + c = 0$** ?
- Duotos dviejų taškų  $(x_1, y_1)$  ir  $(x_2, y_2)$ . Kuris taškas yra daugiau nutolęs nuo x ašies?

# Išreikšta tipų konversija

Sveikąją reikšmę galim priskirti realiajai (netiesioginė konversija), o atvirkščiai – ne. Reikia išreikštos konversijos.

```
x = (int)Math.Pow(y, 0.5);
```

y – sveikojo/realaus tipo.

Toks konversijos formatas, `x = (int)...`, galioja visiems reikšmės (**Value**) tipams (**string** – nėra reikšmės tipas).

```
Console.WriteLine("{0} {1}",  
(int)Math.Sqrt(5.0), (int)Math.Sqrt(8.0)); Koks  
atsakymas?
```

# Varianto operatorius **switch**

Išrinkimas, kai daug variantų.

Išraiškos reikšmė lyginama su pastoviosiomis reikšmėmis.

Tiek išraiškos, tiek konstantų tipai turi sutapti. Jie gali būti sveikojo tipo (visi galimi sveikieji tipai, **char** irgi), eilutės tipo.

# Pavyzdys (1)

```
string kompiuterių_kartos;  
...  
switch (kompiuterių_kartos)  
{  
    case "pirma":  
        Console.WriteLine("Pirma: 1945 - 1958 metai.");  
        break;  
    case "antra":  
        Console.WriteLine("Antra: 1959 - 1963 metai.");  
        break;  
    case "trečia":  
        Console.WriteLine("Trečia: 1964 - 1970 metai.");  
        break;  
    case "ketvirta":  
        Console.WriteLine("Ketvirta: 1971 - dabar.");  
        break;  
    case "penkta":  
        Console.WriteLine("Penkta: nuo dabar į ateitį.");  
        break;  
    default:  
        Console.WriteLine("Tokios kompiuterių kartos nėra.");  
        break;  
}
```

# Pavyzdys (2)

```
switch (p)
{
    case 10:
    case 9:
        q++;
        break;
    case 8:
        q = q + 2;
        break;
    case 7:
        q = q + 3;
        break;
    case 6:
        break;
    default:
        q = q + 4;
        break;
}
```

Koks atsakymas:

➤ p = 10, q = 10?

➤ p = 6, q=10?

➤ p = 20, q = 10?



# *Ciklo operatoriai*

# Ciklo sakiny **while**

**while** (Sąlyga)

{

    //Ciklo kamienas

    . . .

}

i = 1;

**while** (i <= 10)

{

    Console.WriteLine("{0,3:d} {1,6:d}", i, i \* i);

    i++;

}

# Ciklo sakiny **for**(1)

**for** (R1; R2; R3)

**KartojamasSakinys;**

**R1** – pradinių reikšmių suteikimo reiškinys,

**R2** – pabaigos sąlygos tikrinimas (vykdymo reiškinys),

**R3** – kitimo reiškinys.

**KartojamasSakinys** – bet koks C# kalbos sakiny (tame tarpe ir ciklo).



# Ciklo sakiny **for**(2)

```
int x = 10, y;  
for (int i = 1; i < 10; i++){  
    x = 3 * x - 2;  
    y = x * x + x - 1;  
    Console.WriteLine("{0,3:d} {1,6:d}", i, y);  
}
```

Lokalus  
kintamasis

```
for (int j = 1, s = 0; j < 10; s = s + j, j++){  
    Console.WriteLine("{0,3:d} {1,6:d}", j, s);  
}
```

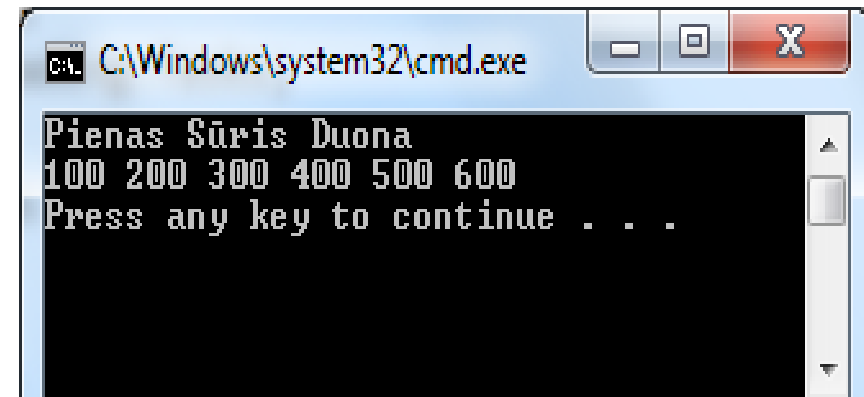
# Ciklo sakiny **foreach**

```
foreach (Kintamojo_tipas kintamasis in konteineris)  
    KartojamasSakiny;
```

```
string [] Prekė = {"Pienas", "Sūris", "Duona"};  
int [] Skaiciai = {100, 200, 300, 400, 500, 600};
```

```
foreach (string pr in Prekė)  
    Console.Write ("{0} ",pr);  
Console.WriteLine();
```

```
foreach (int sk in Skaiciai)  
    Console.Write("{0} ", sk);  
Console.WriteLine();
```



A screenshot of a Windows command prompt window titled "C:\Windows\system32\cmd.exe". The window has standard Windows window controls (minimize, maximize, close) in the top right corner. The command prompt displays the output of the C# program: "Pienas Sūris Duona" on the first line, "100 200 300 400 500 600" on the second line, and "Press any key to continue . . ." on the third line. The text is displayed in a monospaced font on a black background.

# Operatorius **break**

```
int p = 0;
for (int i = 0; i < 10; i++)
{
    p = p + i;
    if (p > 15)
        break;
    Console.Write("{0} ", i);
}
Console.WriteLine(" ");
```



# *Duomenų įvestis ir išvestis (darbas su konsole)*

# Viena reikšmė eilutėje (1)

```
Console.WriteLine("Įveskite teksto eilutę");
```

```
string kint5 = Console.ReadLine();
```

```
Console.WriteLine("Įveskite sveikąjį");
```

```
int kint1 = int.Parse(Console.ReadLine());
```

```
Console.WriteLine("Įveskite realųjį");
```

```
double kint2 = double.Parse(Console.ReadLine());
```

Metodas `Parse(Console.ReadLine())` konvertuoja įvedama eilutę į atitinkamo tipo duomenis. **Realaus kintamojo reikšmę reikia vesti su kableliu!**

# Viena reikšmė eilutėje (2)

```
Console.WriteLine("Įveskite dešimtainį");  
decimal kint3 = decimal.Parse(Console.ReadLine());
```

```
Console.WriteLine("Įveskite simbolių");  
char kint4 = (char)Console.Read();
```

Simbolių įvedimui naudojamas metodas `(char)Console.Read()`. Įvedus simbolių, lieka nepanaudota likusi eilutės dalis (*Enter* klavišo) paspaudimas. Tai reikia įvertinti, jei dar reikės atlikti duomenų įvedimą. Tam papildomai atliekamas likusios eilutės dalies nuskaitymas:

```
string Kint_nereikalingas = Console.ReadLine();
```

# Viena reikšmė eilutėje (3)

```
Console.WriteLine("Įveskite sveikąjį");  
kint1 = Convert.ToInt32(Console.ReadLine());  
Console.WriteLine("Įveskite realųjį");  
kint2 = Convert.ToDouble(Console.ReadLine());  
Console.WriteLine("Įveskite dešimtainį");  
kint3 = Convert.ToDecimal(Console.ReadLine());  
Console.WriteLine("Įveskite simbolių");  
kint4 = Convert.ToChar(Console.ReadLine());  
Console.WriteLine("Įveskite teksto eilutę");  
kint5 = Console.ReadLine();
```

Naudojant metodą `Convert.ToChar(Console.ReadLine())`, simbolių galima įvesti bet kada.

# Daug reikšmių eilutėje

```
Console.Write("Įveskite sveikąjį, realųjį, dešimtainį, ");  
Console.WriteLine("simbolį, eilutę visus vienoje eilutėje");  
string eilute = Console.ReadLine(); // visos eilutės įvedimas  
string[] dalys = eilute.Split(' ');  
// Kintamųjų reikšmės eilutėje skiriamos vienu tarpu
```

```
int sk1 = Convert.ToInt32(dalys[0]);  
double sk2 = Convert.ToDouble(dalys[1]);  
decimal sk3 = Convert.ToDecimal(dalys[2]);  
char sk4 = Convert.ToChar(dalys[3]);  
string sk5 = dalys[4];
```

Skyriklis tarp elementų gali būti bet koks, bet tokio simbolio negali būti **string** tipo eilutėje. Be tarpo simbolio dar naudojamas ; simbolis. Duomenų eilutės pavyzdys: **20 17,6 33,8 ? Darbas**



# Išvedimo formatavimas (1)

Dažniausiai naudojami formatai:

Simbolis	Paskirtis	Pastabos
-	Talpinamas prie kairės	Eilutėms.
D ar d	Sveikieji	
F ar f	Realieji	Galima nurodyti pozicijų skaičių po kablelio, numatytoji 2.
C ar c	Dešimtainiai	Spausdina valiutos ženklą. Konsolėje nematomas.

# Išvedimo formatavimas (2)

```
Console.WriteLine("Sveikasis {0} realusis {1}", sk1, sk2);
```

```
Console.WriteLine(" dešimtainis {0} simbolis {1}  
tekstas {2}", kint1, kint2, kint3); // taip negalima!!!
```

```
Console.WriteLine(" dešimtainis {0} simbolis " +  
"{1} tekstas {2}", sk1, sk2, sk3); // taip skaityti galima
```

```
Console.WriteLine(" skaičius {0} simbolis {1} tekstas {2}",  
kint1, kint2, kint3); // taip skaityti galima
```

# Išvedimo formatavimas(3)

```
int kint11;  
double kint12;  
decimal kint13;  
string kint14, kint15;
```

...

```
Console.WriteLine(" Dydis1={0,4:d} dydis2={1,8:f3} " +  
    " dydis3={2}", kint11, kint12, kint13);  
Console.WriteLine(" {0,-10} {1,12} ", kint14, kint15);
```

Kintamajam skirtos  
8 pozicijos. 3 iš jų –  
trupmeniniai daliai.



# *Klasės sudarymas*

# Klasės kūrimas

```
public class Prekė
{
    // klasės duomenys
    ...
    // klasės metodai
    ...
}
```

# Klasės elementų matomumas

Klasės elementų matomumu vadinama galimybė juos panaudoti klasės viduje ir išorėje.

Klasės elementų matomumas priklauso nuo to, kokie matomumo požymiai jiems nurodyti.

Matomumo požymis suteikiamas užrašius vieną iš raktinių žodžių **private** (privatus), **public** (viešasis).

Savybės dažniausiai turi matomumo požymį **private**, metodai - **public**.

# Požymis **private**

Šis požymis nurodo, kad aprašytą klasės elementą (savybę ar elgseną) gali naudoti tik šios klasės metodai.

Tai **numatytoji** (galiojanti, jei matomumas nenurodytas) klasės elementų matomumo reikšmė.

# Požymis **public**

Klasės elementas matomas (gali būti naudojamas) ir už klasės ribų.

Šį požymį klasės elementai įgauta tik nurodžius tiesiogiai.



# Klasės vardų galiojimo sritis

Klasės viduje aprašyti kintamieji privalo turėti **skirtingus** vardus.

Klasės viduje paskelbti vardai **galioja visoje klasėje**.

Klasės elementai klasės išorėje gali būti naudojami (jų vardai galioja), **jei tai leidžia jų matomumo požymis**.

Klasės išorėje prieš naudojamą klasės elemento vardą turi būti nurodytas **objekto vardas**. Pavyzdžiui, **objektas1.laukovardas**.

# Pirmoji klasė

```
/** Klasė prekės duomenims saugoti
@class Prekė */
public class Prekė
{
    private string pavad;    // prekės pavadinimas
    private double kaina,    // prekės vieneto kaina
                kiekis;      // prekės kiekis

    /** Spausdina pranešimą */
    public void ParodytiPranešimą()
    {
        Console.WriteLine("Klasės Prekė metodas");
    }
}
```

# Pagrindinė klasė, pagrindinis metodas

```
/** Pagrindinė klasė */
```

```
class Program
```

```
{
```

```
    static void Main(string[] args)
```

```
    {
```

```
        Prekė pr1 = new Prekė(); //sukuriamas objektas
```

```
        pr1.ParodytiPranešimą(); //klasės metodo panaudojimas
```

```
    }
```

```
}
```



## *Sąsajos tarp objektų*

# Sąsajų esmė

Skelbti objekto savybes atvirosiomis nerekomenduojama, nes pažeidžiamas kertinis objektinio programavimo principas – *inkapsuliacija*.

Objekto elgsenos aprašyme galima numatyti galimybę keistis duomenimis su aplinka. Tam naudojami specialūs *atvirieji metodai – sąsajos*, duomenų priėmimui ir išdavimui.

# Metodas (1)

Metodu vadinama specialiai apiforminta objekto elgseną nusakanti programos dalis.

Metodo aprašymas talpinamas klasėje ir susideda iš vardo, parametrų sąrašo ir kamieno.

Jei reikia, per parametrus metodui perduodami duomenys.

Pagal rezultatų grąžinimo būdą metodai skirstomi:

- atsakymą grąžinantys per metodo vardą;
- atsakymą grąžinantys per parametrų sąrašą.

# Metodas (2)

Metodo, grąžinančio atsakymą per vardą, bendrasis pavidalas:

**Tipas** MetodoVardas(Parametrai)

{

    //Kamienas

    . . .

**return** atsakymas;

}

Parametrų gali ir nebūti. Būtinasis **return**.

# Metodas (3)

```
public double RastiSumą()  
{  
    double suma;  
    suma = kiekis * kaina;  
    return suma; // grąžina atsakymą  
}
```



# Metodo dalys

**MetodoVardas** – metodo esmę atspindintis kintamasis, naudojamas kreipinyje ir per kurį grąžinamas atsakymas.

**Tipas** – metodo vardo (kintamojo) tipas.

**Parametrai** – kintamųjų (metodo parametrų) aprašymas. Parametrai aprašomi kaip paprasti kintamieji – juos naudosime **Kamiene**.

**Atsakymas** – reiškinys, pagal kurį apskaičiuojama atsakymo reikšmė. Operatorius **return** šią reikšmę suteikia reikšmę kintamajam **MetodoVardas** ir užbaigia metodo darbą.

**Kamienas** – atliekamus veiksmus aprašanti programa.

# Kreipiniai į metodus klasės viduje

Kreipinyje nurodomas metodo, į kurį kreipiamasi, vardas, faktinės parametrų reikšmės (argumentai) ir kintamasis, kuriame įsimenama metode apskaičiuota reikšmė.

Tipinis kreipinys į metodą, kuris atsakymą grąžina per vardą, klasės viduje yra:

**Kintamasis = MetodoVardas (Argumentai);**

Metodo viduje būtinai yra **return**.

# Kreipinio dalys

**Kintamasis** – kintamasis, kuriame įsimenama iš metodo per jo vardą grąžinta reikšmė.

**MetodoVardas** – metodo, į kurį kreipiamasi, vardas.

**Argumentai** – konstantomis, kintamaisiais ar išraiškomis nurodytos reikšmės, kurios prieš atliekant skaičiavimus suteikiamos metodo parametrų sąrašui.

Argumentų skaičius ir tipai turi atitikti parametrų skaičių ir tipus, parametrų sąrašui argumentų reikšmės suteikiamos parametrų sąrašui išvardintos eilės tvarka.

# Išoriniai kreipiniai (1)

Jei į metodą kreipiamasi iš išorės, pavyzdžiui, iš **Main ()** metodo, reikia nurodyti, į kurio objekto metodą kreipiamės. Tai daroma pagal tas pačias taisykles, kaip ir kreipiantis į savybes. Į metodus **Suma()** bei **Vidurkis()**, jei jie būtų aprašyti klasėje, galima kreiptis:

```
int x, y;
```

```
double z, t, p;
```

```
. . .
```

```
y = objektas1.Suma(x, 5);
```

```
z = objektas2.Vidurkis(t, p);
```

# Išoriniai kreipiniai(2)

Kreipiantis iš išorės, pavyzdžiui, iš **Main()** metodo, reikia nurodyti į kurio objekto metodą kreipiamės pagal tas pačias taisykles, kaip ir kreipiantis į savybes. Į metodą

**ŠeimosPajamos()** galima kreiptis:

```
Šeima JonoIrMarytėsŠeima = new Šeima();
```

```
double pirmasAtl, antrasAtl, šeimosUždarbis;
```

...

```
šeimosUždarbis =  
    JonoIrMarytėsŠeima.ŠeimosPajamos  
    (pirmasAtl, antrasAtl);
```

# Tipas **void**

Jei metodui reikšmės grąžinti nereikia, galima naudoti specialų tipą **void**. Tuomet operatoriuje **return** atsakymo išraiška nenurodoma arba jis iš viso nerašomas:

```
public void Linija(int ilgis)  
{  
    ...  
}
```

Kreipiamasi į tokį metodą be priskyrimo operatoriaus:

```
Linija(15) ;
```

# Metodai be parametru

Jei metodui parametru nereikia, aprašyme ir kreipinyje skliaustai išlieka:

```
public void Linija32( )  
{ ...  
}
```

Kreipinys:

```
Linija32( );
```

# Sąsajos metodai (1)

Sąsajos metodais vadinami metodai, skirti paimti arba įrašyti objektų savybių reikšmes.

Jei metodas savybės reikšmę perduoda į išorę, jo pavadinime rašome žodį `Imti` ir prijungiame parametro pavadinimą. Pavyzdžiui, `ImtiKiekį()`, `ImtiKainą()`.

Jei metodas iš išorės gautą reikšmę įrašo į objekto savybės lauką, jo pavadinime rašome žodį `Dėti()` ir prijungiame parametro pavadinimą. Pavyzdžiui, `DėtiKiekį()`, `DėtiKainą()`.



# Sąsajos metodai (2)

Sąsajos metodai turi būti viešieji (**public**).

Sąsajos metodai yra objekto elgseną aprašantys metodai ir aprašomi klasėje.

Objekto savybes keičiantys sąsajos metodai (Dėti...) prieš suteikdami nurodytą reikšmę savybei, paprastai tikrina nurodytos reikšmės teisingumą. Pavyzdžiui, jei tai meniuo, tai ar nurodytas sveikas skaičius iš intervalo 1-12 ir pan. Dėl vietos trūkumo skaidrėse per paskaitas tokių patikrinimų nedarysime.

# Klasės sąsajos Dėti() metodai

```
/** įrašo prekės pavadinimą */
public void DėtiPavadinimą(string pavad)
{
    this.pavad = pavad;
}
```

```
/** įrašo prekės kainą */
public void DėtiKainą(double kaina)
{
    this.kaina = kaina;
}
```

```
/** įrašo prekės kiekį */
public void DėtiKiekį(double kiekis)
{
    this.kiekis = kiekis;
}
```

**this** naudojamas tik tada, kai klasės savybės vardas sutampa su metodo parametro vardu.

# Klasės sąsajos Imti() metodai

```
/** grąžina prekės pavadinimą */
```

```
public string ImtiPavadinimą() { return pavad; }
```

```
/** grąžina prekės kainą */
```

```
public double ImtiKainą() { return kaina; }
```

```
/** grąžina prekės kiekį */
```

```
public double ImtiKiekį() { return kiekis; }
```

# Sąsajos metodų panaudojimas

```
Prekė pr1 = new Prekė(); //sukuriamas objektas
```

```
...
```

```
pr1.DėtiPavadinimą("Knyga");
```

```
pr1.DėtiKainą(20);
```

```
pr1.DėtiKiekį(100);
```

```
Console.WriteLine("{0}    {1,8:f}    {2,8:f}",
```

```
    pr1.ImtiPavadinimą(), pr1.ImtiKainą(),
```

```
    pr1.ImtiKiekį());
```



# *Programavimo stilius, kultūra*

# Programavimo stilius, kultūra

Tikslas – programos aiškumas.

Stilius, kultūra – tai vardų parinkimas, teksto išdėstymas, komentarų naudojimas ir pan.

Taisyklės nėra privalomos (kompiuteris “supras” bet kaip išdėstyta programą), bet rekomenduojama jų laikytis.

Geriau savas stilius, negu jokio.

# Programos išdėstymas faile

Rekomenduojama programos elementų išdėstymo seka:

- **Using** direktyvos
- Konstantos
- Nauji tipai (klasės)
- Vykdomoji klasė





# Metodų užrašymas

Metodo kamieno pradžios skliaustas rašomas atskiroje eilutėje, lygiuojant su antraštės pradžia. Toje eilutėje nieko nerašome.

Pabaigos skliaustas rašomas atskiroje eilutėje, lygiuojant su metodo (funkcijos) antraštės pradžia.

```
static void Vardas()  
{  
    . . .  
    // sakiniai  
}
```

# Blokų ir sakinių išdėstymas

Sakiniai bloke rašomi nuo tos pačios pozicijos.

Operatoriaus atidarantis skliaustas rašomas šalia operatoriaus toje pačioje eilutėje arba atskiroje eilutėje operatoriaus pradžios pozicijoje.

Uždarantis skliaustas lygiuojamas su operatoriaus pavadinimo pozicija:

<b>Operatorius {</b>	arba	<b>Operatorius</b>
...		{
...		...
}		}

# if-else išdėstymas

**else** rašomas po **if**, kad matytųsi priklausomybė:

```
| if (Sąlyga)  
|     Sakinys1;  
| else  
|     Sakinys2;
```

```
if (x > y) {  
    x = x++;  
    y = y--;  
}  
else {  
    x = x--;  
    y = y++;  
}
```

# Ciklų išdėstymas (1)

Vidiniai blokai stumiami į dešinę, užbaigus bloką grįžtama:

```
while (Sąlyga) {  
    ...  
    // sakiniai  
}
```

```
for (R1; R2; R3) {  
    ...  
    // sakiniai  
}
```

# Ciklų išdėstymas (2)

```

kiek = 0;
suma = 0.0;
int amžius;
for (int i = 0; i < n; i++)
{
    amžius = metai - D[i].ImtiMetus();
    if ((amPr <= amžius) && (amžius <= amPb))
    {
        suma = suma + D[i].ImtiKainą();
        kiek++;
    }
}

```

# Vardų parinkimas (1)

Vardai turi atspindėti kintamojo (konstantos, metodo) paskirtį, pvz., **atstumas**.

Masyvų, klasių, metodų vardus pradėkite didžiąja raide, pvz., **Masyvas**.

Kintamųjų vardus pradėkite mažąja raide, pvz., **butas**, **butoPlotas**.

Sudėtiniuose varduose antrą ir kitus žodžius pradėkite didžiąja raide, pvz., **namoPlotas**, **butoPlotas**.

Konstantų vardus pradėkite **C** raide, pvz., **CMaxK**.

# Vardų parinkimas (2)

Metodų, kurių tipas yra `void`, vardus pradėkite veiksmazodžio bendratimi, pvz., `RastiNamoPlota()`;

Metodus, kurios skaičiuoja ir grąžina vieną reikšmę, vadinkite tą reikšmę atspindinčiu vardu, pvz., `NamoPlotas()`;

# Komentarai skirti:

- Nurodyti bendrą programos paskirtį
- Nurodyti konstantų paskirtį
- Nurodyti klasių paskirtį
- Nurodyti klasių kintamųjų paskirtį
- Nurodyti pagrindinių programos kintamųjų paskirtį
- Aiškinti kiekvieno metodo:
  - ✓ paskirtį
  - ✓ naudojamus parametrus
  - ✓ grąžinamą reikšmę
- Aiškinti ypatingų programos vietų ir svarbiausių blokų paskirtį



# Programos paskirtis

```
// Ši programa randa visus nurodyto katalogo failus  
// ir jų duomenis įveda į programą.  
// Autorius: Vardenis Pavardenis  
using System;  
using System.IO;  
using System.Collections.Generic;  
using System.Linq;  
using System.Text;  
using System.Threading.Tasks;
```

# Konstantų paskirtis

```
const string CFd = "..\\..\\Duomenys.txt";    // Pirmo duomenų failo vardas
const string CFd1 = "..\\..\\Duomenys1.txt"; // Antro duomenų failo vardas
const string CFr = "..\\..\\Atsakymai.txt";   // Atsakymų failo vardas
const int maxKiekis = 100;                   // Maksimalus elementų kiekis masyve
```

# Klasių paskirtis

```
/** Klasė mokinio duomenims saugoti  
@class Mokinys */  
class Mokinys  
{  
    . . .  
}
```

# Klasės kintamųjų paskirtis

```
private string pav,      // mokinio pavardė
                    vard; // mokinio vardas
private int klas,       // klasė
                    laikas; // laikas, praleistas internete
private double vid;     // mokymosi vidurkis
```

# Pagrindinio metodo kintamųjų paskirtis

```
static void Main(string[] args)
{
    Dviratis [] D = new Dviratis[Cn]; // dviračių duomenys -
                                        // objektai
                                        // Cn - max objektų kiekis

    int n;      // dviračių skaičius
    int am;     // dviračio tinkamumo naudoti kritinis amžius
    . . .
}
```

# Aiškinamas metodas

**Paaiškinama metodo paskirtis, parametrai, grąžinama reikšmė:**

```
//-----  
/** Skaičiuoja ir grąžina nurodyto dviračių amžiaus intervalo vidurkį  
@param D - dviračių duomenys  
@param n - dviračių skaičius  
@param metai - metai, kurių atžvilgiu skaičiuojamas dviračio amžius  
@param amPr - dviračių paieškos amžiaus intervalo pradžia  
@param amPb - dviračių paieškos amžiaus intervalo pabaiga */  
//-----  
static double Vidurkis(Dviratis[] D, int n, int metai, int amPr, int amPb)  
{  
    ...  
}
```

# Aiškinamos ypatingos vietos, blokų paskirtis

```
for (int i = 0; i < kd; i++){  
    if (K[i] == numeris){  
        atsakymas = K[i];  
        i = i + kd;    // suradus atsakymą, nutraukiamas ciklas  
    }  
}  
  
// Norimų metų (2010) dviračių kainų sumos ir vidutinio amžiaus radimas  
double sumaTinka2;  
Pinigai(D, n, 0, 0, 2010, out kiekTinka, out sumaTinka2);  
if (sumaTinka2 > 0) {  
    double vidurkisTinka2 = Vidurkis(D, n, 2015, 0, 2015 - 2010);  
    Console.WriteLine("Norimų metų (2010) dviračių kainų suma: {0,7:f2}\n"  
        + "Jų vidutinis amžius: {1,7:f2}\n",  
        sumaTinka2, vidurkisTinka2);  
}
```



*Klausimai?*