

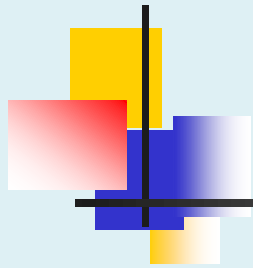
# **T2. Programos struktūrizācija**



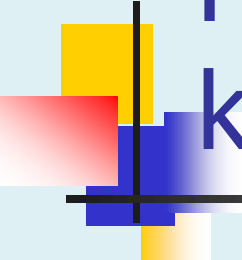
# Temos klausimai

---

- Funkcija, kaip bazinė JAVA konstrukcija. Funkcijų paskelbimas ir iškvietimas.
- Funkcijos parametrai. Duomenų perdavimas funkcijai.
- Funkcijos grąžinama reikšmė. Kreipiniai į funkciją.
- Loginės operacijos. Loginis kintamasis (bool). Loginis reiškiny.
- Programavimo stilius, kultūra. Komentarų naudojimas programoje.



# **JAVA funkcijos**



# Funkcija – bazinė JAVA konstrukcija

---

- Sudaroma kokiam nors vienam konkrečiam veiksmui atlikti.
- Suteikia galimybę struktūrizuoti programą – padalyti programos tekstą į mažus lengvai suprantamus gabalėlius.
- Suteikia galimybę pasikartojančius veiksmus su skirtingais duomenimis sujungti į vieną funkciją.



# Funkcijos užrašymo tvarka

---

- Tekstas užrašomas už pagrindinės programos (main funkcijos):

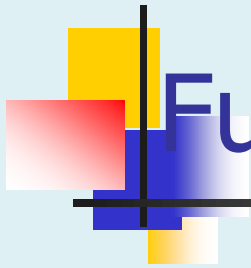
**Tipas Vardas(Parametrų sąrašas)**

**{**

**veiksmai**

**}**

- Funkcija panaudojama, jei kitoje funkcijoje ar pagrindinėje programoje yra kreipinių į tą funkciją.



# Funkcijos grąžinamos reikšmės tipas

---

- Bet kuris standartinis JAVA tipas.
- Bet kuris programuotojo anksčiau aprašytas tipas.
- **void**.



# Funkcijos parametrai

---

- Paskirtis – perduoti funkcijai duomenų reikšmes ir grąžinti rezultatus.
- Parametrų sąrašas funkcijos teksto antraštėje:  
**Tipas1 Vardas1, Tipas2 Vardas2, ...**
- Jei parametrų nėra, naudojami tušti skliaustai.



# Funkcijų kreipiniai

---

- Kreipinys į funkciją:  
**Vardas (Argumentų sąrašas);**
- Argumentų sąrašas:  
**Arg1, Arg2, ...**
- Argumentų sąrašas turi atitikti parametrų sąrašą.





# Duomenų perdavimas funkcijai

---

- Per funkcijos kreipinio argumentus.
- Per globalius programos kintamuosius (**nerekomenduotina**).



# Funkcijos grąžinamos reikšmės tipo parinkimas

---

- Funkcijos grąžinamos reikšmės tipas, užrašytas antraštėje prieš funkcijos vardą, – tai per funkcijos vardą grąžinamos reikšmės tipas.
- Jei per vardą nieko negrąžinama, funkcijos grąžinamos reikšmės tipas turi būti **void**.



# Funkcijos rezultatas

---

- Reikšmei grąžinti per funkcijos vardą naudojamasis sakinyss:  
**return išraiška;**
- Grąžinamo reiškinio tipas turi atitikti funkcijos grąžinamos reikšmės tipą.
- Funkcijoje gali būti keli **return** sakiniai.
- Įvykdžius **return** sakinį, išeinama iš funkcijos.



# Funkcijų naudojimas

---

- Kreipinys į funkciją, kurios tipas **void**, turi būti užrašytas atskiru sakiniu:  
**vardas (argumentų sąrašas);**
- Kreipinys į funkciją, grąžinančią reikšmę, turi dalyvauti reiškinyje, pvz.:  
**kint = vardas (argumentų sąrašas);**

# 1-as pavyzdys: funkcija, grąžinanti reikšmę

```
public class Suma {  
    public static void main(String args[])  
    {  
        double a = 3.2, b = 8.6;  
        double c = sudeti(a, b);  
    }  
    // -----  
    // Funkcija dviejų skaičių sumai rasti  
    static double sudeti(double x, double y)  
    {  
        double z = x + y;  
        return z;  
    }  
} //class
```

Kreipinys į  
funkciją

Funkcijos  
antraštė

Funkcijos  
kamienas

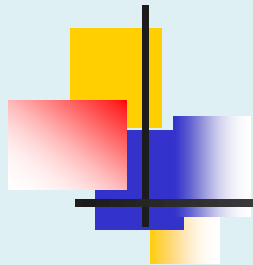
## 2-as pavyzdys: funkcija, negražinanti reikšmės

```
public class Linija {  
    public static void main(String args[])  
    {  
        breztiLinija(20, '-');  
    }  
    // -----  
    // Funkcija duoto ilgio linijai brėžti  
    // simb - linijos simbolis  
    static void breztiLinija(int ilgis,  
    char simb)  
    {  
        for (int i=0; i<ilgis; i++)  
            System.out.print(simb);  
        System.out.println();  
    }  
} //class
```

Kreipinys į  
funkciją

Funkcijos  
antraštė

Funkcijos  
kamienas



# **Loginės operacijos ir loginiai reiškiniai**



# Loginis kintamasis

---

- Loginis kintamasis:  
`bool vardas;`
- Galimos reikšmės:  
`true (≠0), false (0)`
- Pavyzdys:  
`bool a = true, b = false;`





# Loginės operacijos

---

- Loginė daugyba (ir):  
&&
- Loginė sudėtis (arba):  
||
- Loginis neigimas (ne):  
!



# Loginių operacijų rezultatai

---

Veiksmas	Rezultatas
false && false false && true true && true	false false true
false    false false    true true    true	false true true
!false !true	true false

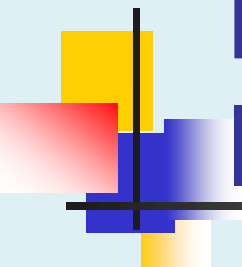


# Loginiai reiškiniai

---

- Sudaromi, sujungiant loginius operandus loginių operacijų ženklais, gali būti naudojami skliausteliai.
- Loginiai operandai:
  - loginės reikšmės,
  - loginiai kintamieji,
  - santykio reiškiniai.
- Pavyzdys:

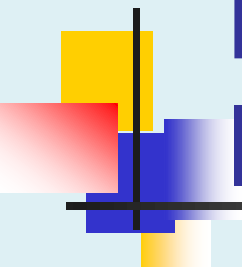
```
int a, b; a = ...; a = ...;  
bool c = a > 0 && b > 0;
```



# Dažnai naudojami loginiai reiškiniai (1)

---

- $x \in [a, b]$  (priklauso intervalui):  
 $a \leq x \ \&\& \ x \leq b$   
arba  
 $!(x < a \ || \ x > b)$
- $x \notin [a, b]$  (nepriklauso intervalui):  
 $x < a \ || \ x > b$   
arba  
 $!(a \leq x \ \&\& \ x \leq b)$



# Dažnai naudojami loginiai reiškiniai (2)

---

- $x \in [a, b]$  ir  $x \in [c, d]$  (priklauso intervalų sankirtai):  
 $(a \leq x \ \&\& \ x \leq b) \ \&\& \ (c \leq x \ \&\& \ x \leq d)$
- $x \in [a, b]$  arba  $x \in [c, d]$  (priklauso bent vienam iš intervalų):  
 $(a \leq x \ \&\& \ x \leq b) \ || \ (c \leq x \ \&\& \ x \leq d)$



---

# Programavimo stilius



# 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.



# Funkcijų užrašymas

---

- Funkcijos kamieno pradžios skliaustas rašomas atskiroje eilutėje, lygiuojant su antraštės pradžia. Toje eilutėje nieko nerašome.
- Pabaigos skliaustas rašomas atskiroje eilutėje, lygiuojant su funkcijos antraštės pradžia.

```
void rasti()
```

```
{
```

```
    sakiniai;
```

```
}
```





# Blokų ir sakinių išdėstymas

---

Blokai, sakiniai bloke – nuo tos pačios pozicijos.  
Operatoriaus atidarantis skliaustas rašomas šalia  
operatoriaus toje pačioje arba atskiroje eilutėje.  
Uždarantis skliaustas lygiuojamas su operatoriaus  
pavadinimo pozicija:

<pre>Operatorius {     . . .     . . . }</pre>	arba	<pre>Operatorius {     . . . }</pre>
--	------	--



# if-else išdėstymas

---

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

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



# Ciklų išdėstymas

---

Vidiniai blokai stumiami į dešinę, užbaigus bloką grįžtama:

```
while (Sąlyga) {  
    ...  
    Sakiniai;  
}
```



# Vardų parinkimas

---

- Vardai turi atspindėti kintamojo paskirtį, pvz., **atstumas, rezultatas**.
- Funkcijų vardai (veiksmažodio bendratis) iš mažosios raidės, pvz., **sudeti, atimti**.
- Sudėtiniuose varduose antrą žodį pradėkite didžiąja raide, pvz., **namoPlotas**.
- Klasių vardai (daiktavardis, vns) iš didžiosios raidės, pvz., **Suma, Linija**.
- Masyvai (dgs) iš mažosios, pvz., **skaiciai**.



# Komentarų naudojimas programoje

---

- Nurodyti bendrą programos paskirtį.
- Nurodyti konstantų paskirtį.
- Nurodyti pagrindinių programos kintamųjų paskirtį.
- Aiškinti kiekvienos funkcijos paskirtį ir naudojamus parametrus.
- Aiškinti ypatingų programos vietų ir svarbiausių blokų paskirtį.



# Code Style for Contributors

---

The code styles below are **strict rules**, not guidelines or recommendations.

**We recognize that not all existing code follows these rules, but we expect all new code to be compliant:**

<http://www.oracle.com/technetwork/java/codeconventions-150003.pdf>



# Sužinojome

---

**Kaip programuoti JAVA funkcijas.**

**Kaip naudoti loginius kintamuosius ir logines operacijas.**

**Koks programos rašymo stilius gali būti laikomas geru programavimo stiliumi.**