

JAVA: while|do-while|for

Mindaugas Karpinskas
2017



Trumai

- if (boolean)
- else if (boolean)
- else (boolean)
- switch (boolean)
- **while** (boolean)
- **for** (... , (boolean), ...)
- **do** {...} **while** (boolean)
- **break**, **continue** ir **return**

<https://www.geeksforgeeks.org/loops-in-java/>

IF - IF ELSE

prisminkime

if , else if, else

```
if (done == true) // arba if (done)
    System.out.println("atlikau");
else
    System.out.println("tęsiu");
```

.....

```
int temp;
if (temp < 0)
    System.out.println("Jo, šiandien šaltoka.");
else if (temp > 100)
    System.out.println("Vanduo jau verda?");
else
    System.out.println("Gera šiandien diena!");
```

Po **if** einanti loginė išraiška yra apskaičiuojama ir gaunamas loginis rezultatas **true** arba **false**.

Rezultatas true priverčia vykdyti komandą stovinčią po **if** , o **false** atveju bus vykdoma po **else** parašyta komanda.

if , else if, else

```
boolean done = true;
if (done == true) { // arba if (done)
    System.out.println("atlikau");
    intSk++;
} else {
    System.out.println("tęsiu");
    intSk++;
}.....
int temp = 100;
if (temp < 0) {
    System.out.println("Jo, šiandien šaltoka.");
} else if (temp > 100) {
    System.out.println("Vanduo jau verda?");
} else {
    System.out.println("Gera šiandien diena!");
}
```

Bloku vadinama grupė komandų, kurios patalpintos tarp riestinių skliaustų poros {}.

if , else if, else

```
int prekėsKaina = 10, turiuPinigų = 20;
```

```
boolean šaliaNėraŽmonos = false;
```

```
if (prekėsKaina <= turiuPinigų)
```

```
    if (šaliaNėraŽmonos == true)
```

```
        System.out.println("Pirksiu");
```

```
    else
```

```
        System.out.println("Nepakanka pinigų");
```

if , else if, else

```
int prekėsKaina = 10, turiuPinigų = 20;
boolean šaliaNėraŽmonos = false;

if (prekėsKaina <= turiuPinigų) {
    if (šaliaNėraŽmonos == true) {
        System.out.println("Pirksiu");
    } else {
        System.out.println("Nepakanka pinigų");
    }
}
```


Ciklai

- While
- Do-while
- For

while (boolean)

```
boolean sąlygosRezultatas = true arba false;  
while( sąlygosRezultatas ) {  
  
}
```

while (boolean)

Sintaksė:

```
while (sąlyga) {  
    Sakiniai;  
    Sakiniai;  
}
```

while (boolean)

Sintaksė:

```
while (sąlyga) {  
    Sakiniai;  
    Sakiniai;  
}
```

- Sakiniai vykdomi tik tuo atveju, jei tenkinama sąlyga
- Jei sąlyga netenkinama iš pat pradžių, sakiniai nevykdomi nei karto
- Sakinių bloke turi būti užtikrinta, kad kada nors sąlyga nebus tenkinama
- Sakinių bloke gali būti ciklo sakinių

while (boolean)

Sintaksė:

```
while (sąlyga) {  
    Sakiniai;  
    Sakiniai;  
}
```

Palyginimo išraiška apskaičiuojama prieš ciklo operatoriaus įvykdymą. Kada ši išraiška yra false , ciklas užbaigiamas. Kad negauti be galo vykdomą ciklą, jūs turite pakeisti loginės išraiškos kintamųjų reikšmes ciklo kamieno viduje.

Sakinio **while** pavyzdys

```
int x = 0;  
while (x < 5) {  
    System.out.println("Mano skaičius: " + x);  
    x++;  
}
```

Sakinio **while** pavyzdys

```
// Kiek sekos 1/n narių reikia susumuoti,  
// kad gautume daugiau už  
float suma = 0;  
int i = 1;  
while (suma < 5) {  
    suma += (float) 1 / i;  
    i++;  
}  
System.out.println(i);
```

Sakinio **while** pavyzdys

```
// Kiek sekos 1/n narių reikia susumuoti,  
// kad gautume daugiau už  
float suma = 0;  
int i = 1;  
while (suma < 5) {  
    suma += (float) 1 / i;  
    i++;  
}  
System.out.println(i); // atsakymas: 84
```

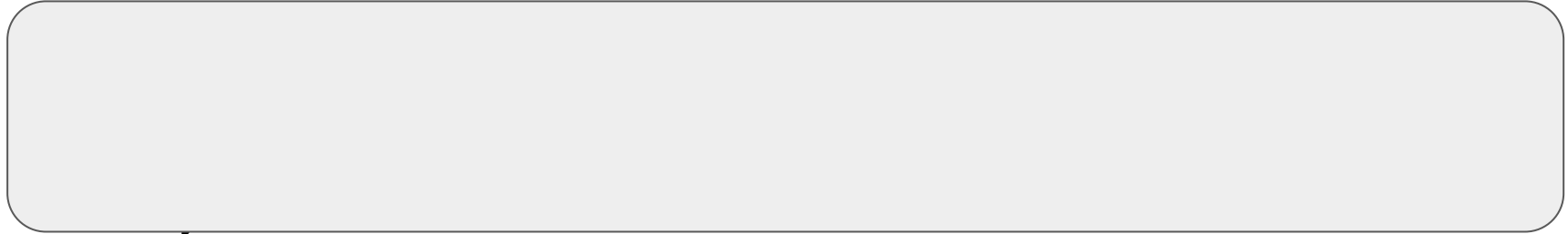

while: Uzduotis1

- Išspausdinti 20 atsitiktinai sugeneruotu int skaičių.
 - Pasinaudot: `random.nextInt(1000);`
- **Būtina:** panaudoti **while** ciklą!
- Taip pat raskite didžiausią sugeneruotą skaičių ir jį išspausdinkite apačioje

PS: reikalingi 3-4 kintamieji

while: Uzduotis1

```
public class Uzduotis1 {  
    public static void main(String[] args) {  
        Random random = new Random();  
        int i = 0;
```



```
    }  
}
```

while: Uzduotis1

```
public class Uzduotis1 {  
    public static void main(String[] args) {  
        Random random = new Random();  
        int i = 0;  
        while (i < 20) {  
            i++;  
            System.out.println(i + " badymas: " + random.nextInt());  
        }  
    }  
}
```

while: Uždutis1

```
public static void main(String[] args) {  
    Random random = new Random();  
    int i = 0;  
    int max = -1;  
    while (i < 20) {  
        i++;  
        int sugeneruotas = random.nextInt(1000);  
        System.out.println(i + " bandomas: " + sugeneruotas);  
        if (sugeneruotas > max) {  
            max = sugeneruotas;  
        }  
    }  
    System.out.println("Sugeneruotas didžiausias skaičius: " + max);  
}
```

while: Uždutis1

```
public static void main(String[] args) {
    Random random = new Random();
    int i = 0;
    int max = -1;
    while (i < 20) {
        i++;
        int sugeneruotas = random.nextInt(1000);
        System.out.println(i + " badymas: " + sugeneruotas);
        if (sugeneruotas > max) {
            max = sugeneruotas;
        }
    }
    System.out.println("Sugeneruotas dydžiausias skaičius: " + max);
}
```

while: Uzduotis2

- Programa turi paprašyti vartotojo įvesti sakinį (tekstinę eilutę). Vartotojui įvedus tekstą programa:
 - Apskaičiuoja iš kelių simboliu sudarytas sakiny
 - Atspausdina paskuti žodį
- Aukščiau išvardintus veiksmus kartoti tok kol įvestas sakiny NEllygus “Viso gero” ty programa vėl prašo įvesti sakinį ir atlieka tuos pačius veiksmus

while: Uzduotis2

```
import java.util.Scanner;  
public class Uzduotis2 {  
    public static void main(String[] args) {  
        Scanner scanner = new Scanner(System.in);  
        String sakinyys = "";
```

```
}
```

```
}
```

while: Uzduotis2

```
import java.util.Scanner;
public class Uzduotis2 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        String sakiny = "";
        while (!sakiny.equals("Viso gero")) {
            System.out.println("Įveskite sakinį:");
            sakiny = scanner.nextLine();
            System.out.println("Sakinys sudarytas iš " + sakiny.length() + " simbolių");
            System.out.println("Sakinio paskutinis šodis "
                               + sakiny.substring(sakiny.lastIndexOf(' ') + 1));
        }
    }
}
```


while: Uzduotis3

- Programa turi paprašyti vartotojo įvesti sakinį (tekstinę eilutę). Vartotojui įvedus tekstą programa:
 - Apskaičiuoja iš kelių simboliu sudarytas sakiny
 - Atspausdina paskutini žodį
 - **Apskaičiuoja kiek sakinyje žodžių ir juos išspausdina**
 - Aukščiau išvardintus veiksmus kartoti tok kol įvestas sakiny NElygus “Viso gero” ty programa vėl prašo įvesti sakinį ir atlieka tuos pačius veiksmus

while: Uzduotis3 based on Uzduotis2

```
import java.util.Scanner;
public class Uzduotis2 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        String sakiny = "";
        while (!sakiny.equals("Viso gero")) {
            System.out.println("\u013veskite sakinj:");
            sakiny = scanner.nextLine();
            System.out.println("Sakiny sudarytas i\u0161 " + sakiny.length() + " simboli\u0177");
            System.out.println("Sakinio paskutinis \u0161odis "
                                + sakiny.substring(sakiny.lastIndexOf(' ') + 1));
            // naujas kodas
        }
    }
}
```

while: Uzduotis3 based on Uzduotis2

```
import java.util.Scanner;
public class Uzduotis2 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        String sakiny = "";
        while (!sakiny.equals("Viso gero")) {
            System.out.println("\u013veskite sakinj:");
            sakiny = scanner.nextLine();
            System.out.println("Sakiny sudarytas i\u0161 " + sakiny.length() + " simboli\u0177");
            System.out.println("Sakinio paskutinis \u0161odis "
                                + sakiny.substring(sakiny.lastIndexOf(' ') + 1));
            // naujas kodas
        }
    }
}
```

while: Uzduotis3

```
while (!sakinys.equals("Viso gero")) {  
    <..>  
    boolean arDarYraTarpu = true;  
    int kiekZodziu = 0;  
    int pirmoTarpoPoz = -1;  
    int antroTarpoPozicija = -1;  
    while (arDarYraTarpu) {  
        kiekZodziu++;  
        pirmoTarpoPoz = antroTarpoPozicija + 1;  
        antroTarpoPozicija = sakinys.indexOf(' ', pirmoTarpoPoz);  
  
    }  
}
```

while: Uzduotis3

```
while (!sakinys.equals("Viso gero")) {  
    <..>  
    boolean arDarYraTarpu = true;  
    int kiekZodziu = 0;  
    int pirmoTarpoPoz = -1;  
    int antroTarpoPozicija = -1;  
    while (arDarYraTarpu) {  
        kiekZodziu++;  
        pirmoTarpoPoz = antroTarpoPozicija + 1;  
        antroTarpoPozicija = sakinys.indexOf(' ', pirmoTarpoPoz);  
        if (antroTarpoPozicija == -1) {  
            arDarYraTarpu = false;  
            antroTarpoPozicija = sakinys.length();  
        }  
        System.out.println(sakinys.substring(pirmoTarpoPoz, antroTarpoPozicija));  
    }  
    System.out.println("Sakinyje rasta " + kiekZodziu + " žodžiai.");  
}
```

while (boolean) - trūkumai

- Jeigu sąlyga pačioje pradžioje grąžina **false**, ciklas **nebus** įvykdytas nei karto.

```
<..>
while (arDarYraTarpu) {
    <..>
    if (antroTarpoPozicija == -1) {
        arDarYraTarpu = false;
    }
}
```

- Jeigu mes norime, kad ciklo kamienas būtų vykdomas mažiausiai kartą, galime naudoti **do-while** ciklą.

do-while

Ciklo sakinyss do-while

Sintaksė:

```
do {
```

```
Sakiniai;
```

```
} while (sąlyga);
```

Sakiniai visada įvykdomi bent vieną kartą

Ciklo sakiny's do-while (boolean) PVZ


do {

// vykdomos komandos

} **while** (sąlyga);

Ciklo sakinyas do-while (boolean) PVZ

```
import java.util.Scanner;
public class PVZ {
    static Scanner scanner = new Scanner(System.in);
    public static void main(String[] args) {
        int suma = 0;
        do {
            System.out.println("Įveskite skaičių: ");
            suma += scanner.nextInt();
        } while (suma < 20);
    }
}
```



do-while: Uzduotis4

- Išspausdinti 20 atsitiktinai sugeneruotu int skaičių.
 - Pasinaudot: `random.nextInt(1000);`
- Taip pat raskite didžiausią sugeneruotą skaičių ir jį išspausdinkite apačioje
- Panaudoti: **do-while**

Uzduotis4

```
import java.util.Random;
public class Uzduotis4 {

    public static void main(String[] args) {
        Random random = new Random();
        int i = 0;
    }
}
```

Uzduotis4

```
import java.util.Random;
public class Uzduotis4 {

    public static void main(String[] args) {
        Random random = new Random();
        int i = 0;
        do {
            i++;
            System.out.println(i + " badymas: " + random.nextInt());
        } while (i < 20);
    }
}
```

do-while: Uzduotis5

- Paprašyti vartotojo įvesti lyginį skaičių. Patikrinti ar tikrai vartotojas įvedė lyginį skaičių. Jei vartotojas visgi įvestu ne lyginį skaičių, prašyti įvesta dar kartą, ir dar kartą..., tol kol įvestas skaičius bus lyginis.
- Išspausdinti įvesta skaičių
- Panaudoti **do-while**

do-while: Uzduotis5

```
import java.util.Scanner;
public class Uzduotis5 {
    static Scanner scanner = new Scanner(System.in);

    public static void main(String[] args) {
        int sk;
        do {
            System.out.println("Lyginis: " + sk);
        }
    }
}
```

do-while: Uzduotis5

```
import java.util.Scanner;
public class Uzduotis5 {
    static Scanner scanner = new Scanner(System.in);

    public static void main(String[] args) {
        int sk;
        do {
            System.out.println("Prašome įvesti lygini skaičių");
            sk = scanner.nextInt();
        } while (sk % 2 != 0);
        System.out.println("Lyginis: " + sk);
    }
}
```


do-while: Uzduotis6 Šilta - Šalta *

- Programa sugeneruoja atsitiktinį skaičių nuo 0 iki 100.
- Vartotojo prašoma atspėti koks sugeneruotas skaičius:
 - Jei skaičius didesnis už įvestą, išveda pranešimą: Didesnis
 - Jei mažesnis, - išveda pranešimą: Mažesnis
- Panaudoti: do-while(sąlyga)

do-while: Uzduotis6 Šilta - Šalta

```
static Scanner scanner = new Scanner(System.in);
static Random r = new Random();
public static void main(String[] args) {
    int atspetiSk = r.nextInt(100);
    int iverstasSk;
    System.out.println("Šilta šalta");
    do {
```

```
        System.out.println("Atspējote " + iverstasSk);
```

```
    }
```

Ciklo sakinyys **for**

Sintaksė:

```
for(inicializavimas; sąlyga; žingsnis) {
```

```
Sakiniai;
```

```
}
```

Po `for` operatoriaus ciklo išraiškos pradžioje yra **inicijuojamas ciklo kintamasis** (skaitliukas), toliau eina **palyginimo** išraiška ir tuomet ciklo **kintamojo keitimo** išraiška.

Ciklo sakinyss for PVZ

1. **for** (**int** i = 0; i < 5; i++) {
 System.**out**.println(i);
 }

2. **int** suma = 0;
 for (**int** i = 0; i < 4; i++) {
 suma += i;
 }

Tipiniu atveju ciklo
kintamasis (skaitliukas) yra
padidinamas arba
sumažinamas vienetu

Ciklo sakinyss for(inicializavimas; sąlyga; žingsnis)

Trys ciklo išraiškos apskaičiuojamos skirtingais būdais.

1. Pirma išraiška (inicijavimo segmentas) įvykdomas vieną kartą pačioje ciklo vykdymo pradžioje.
2. Antroji išraiškos tikrinimo dalis atliekama cikliška kiekvienos ciklo iteracijos pradžioje įskaitant ir patį pirmąjį kartą.
3. Paskutinė trečioji dalis įvykdoma po ciklo kamieno užbaigimo; šioje dalyje standartiniu atveju ciklo išraiškos kintamasis padidinamas arba sumažinamas vienetu.

a. **for** (**int** i = 0; i < 5; i++) {
b. System.**out**.println(i);
c. }

Ciklo sakiny **for**

1. **for** (**int** a = 0, b = 5, c = 50; a < 5; a++) {

}
2. **int** a = 0, b = 5, c = 50;
for (; a < 5; a++) {

}
3. **for** (**int** a = 0, b = 5, c = 50; a < 5; a++, System.**out**.println("Didinam")) {
 System.**out**.println("Spausdinam");
}

Ciklo sakiny **for**

```
public class Pvz4 {  
    public static void main(String[] args) {  
        for (int i = 0, a = a(); b() || i < 1; c(), i++) {  
            d();  
        }  
    }  
    static private int a() {  
        System.out.println("A");  
        return 0;  
    }  
    static private boolean b() {  
        System.out.println("B");  
        return false;  
    }  
    static private boolean c() {  
        System.out.println("C");  
        return false;  
    }  
    static private void d() {  
        System.out.println("D");  
    }  
}
```

Ciklo sakinyys for

A

B

D

C

B

```
public class Pvz4 {  
    public static void main(String[] args) {  
        for (int i = 0, a = a(); b() || i < 1; c(), i++) {  
            d();  
        }  
    }  
    static private int a() {  
        System.out.println("A");  
        return 0;  
    }  
    static private boolean b() {  
        System.out.println("B");  
        return false;  
    }  
    static private boolean c() {  
        System.out.println("C");  
        return false;  
    }  
    static private void d() {  
        System.out.println("D");  
    }  
}
```


Ciklo sakinyss **for**

```
// ....
```

```
for ( ; cont == true; ) {
```

```
    // Komandos kurios atlieka kokius nors veiksmus
```

```
    // ...
```

```
    // Naujos cont reikšmės apskaičiavimo komandos
```

```
    // ...
```

```
}
```

Ciklo sakinyys **for**

```
// .... boolean cont = true;
```

```
for ( ; cont ; ){  
    // Komandos kurios atlieka kokius nors veiksmus  
    // ...  
    // Naujos cont reikšmės apskaičiavimo komandos  
    // ...  
}
```

Masyvai ir **for (...;...;...) { ... }**

```
String[] masyvas = new String[100];  
for (int i = 0; i < 100; i++) {  
    masyvas[i] = "Pradinė reikšmė";  
}
```

for dažnai naudojamas kai iš anksto žinomas ciklo skaičius

for: Uzduotis7 * buvo ND (tik aptarti)

- Paprašyti vartotjo įvesti 10 kartų skaičių nuo 1 iki 5 ir paskaičiuoti kiek iš įvestų skaičių yra: vienetų, dvejetų, trejetų, ketvertų, penketų
 - pasinaudoti: `scanner.nextInt()`;
 - pasinaudoti: `int[]` skaičiai;
 - pasinaudoti: `for`

for: Uzduotis7

```
public static void main(String[] args) {  
    int[] skaiciai = new int[5];
```

```
}
```

for: Uzduotis7

```
public static void main(String[] args) {  
    int[] skaiciai = new int[5];  
    for (int i = 1; i <= 10; i++) {  
        System.out.println(i + " įveskite 1-5 skaičių:");  
        int iverstas = scanner.nextInt();  
        if (iverstas > 0 && iverstas <= 5) {  
            skaiciai[iverstas - 1]++; // skaiciai[iverstas] = skaiciai[iverstas] + 1;  
        }  
    }  
}
```

```
}
```

for: Uzduotis7

```
public static void main(String[] args) {  
    int[] skaiciai = new int[5];  
    for (int i = 1; i <= 10; i++) {  
        System.out.println(i + " įveskite 1-5 skaičių:");  
        int iverstas = scanner.nextInt();  
        if (iverstas > 0 && iverstas <= 5) {  
            skaiciai[iverstas - 1]++; // skaiciai[iverstas] = skaiciai[iverstas] + 1;  
        }  
    }  
  
    for (int i = 0; i < skaiciai.length; i++) {  
        System.out.println("Suvesta " + (i + 1) + " skaičių: " + skaiciai[i]);  
    }  
}
```

ciklai: Uzduotis8*** daryta anksčiau

- Patikrinti ar atsitiktinių skaičių generavimas vienodai atsitiktinai generuoja skaičius nuo 0 iki 9.
 - pasinaudoti: `random.nextInt(10);`
 - pasinaudoti: `int[]` skaičiai;
 - pasinaudoti: `for(`
- Atlikti bandymus
 - 20 kartų,
 - 100 kartų,
 - 10 000 kartų
 - 1000 000 kartų

ciklai: Uzduotis8

```
static Random random = new Random();
```

```
public static void main(String[] args) {
```

```
    int[] skaiciai = new int[10];
```

```
    for (int i = 1; i <= 10; i++)
```

```
}
```

ciklai: Uzduotis8

```
static Random random = new Random();
```

```
public static void main(String[] args) {  
    int[] skaiciai = new int[10];  
    for (int i = 1; i <= 100; i++) {  
        int atsitiktinisSkaicius = random.nextInt(10);  
        skaiciai[atsitiktinisSkaicius]++;  
    }  
    for (int i = 0; i < skaiciai.length; i++) {  
        System.out.println("Sugeneruota " + (i) + " skaičių: " + skaiciai[i]);  
    }  
}
```

ciklai: Uzduotis8 - rezultatai

20x

100x

10 00x

1000 000x

Sugeneruota 1 skaičių: 2
Sugeneruota 2 skaičių: 0
Sugeneruota 3 skaičių: 2
Sugeneruota 4 skaičių: 2
Sugeneruota 5 skaičių: 2
Sugeneruota 6 skaičių: 3
Sugeneruota 7 skaičių: 2
Sugeneruota 8 skaičių: 3
Sugeneruota 9 skaičių: 1
Sugeneruota 10 skaičių: 3

Sugeneruota 1 skaičių: 14
Sugeneruota 2 skaičių: 9
Sugeneruota 3 skaičių: 11
Sugeneruota 4 skaičių: 11
Sugeneruota 5 skaičių: 10
Sugeneruota 6 skaičių: 9
Sugeneruota 7 skaičių: 12
Sugeneruota 8 skaičių: 5
Sugeneruota 9 skaičių: 9
Sugeneruota 10 skaičių: 10

Sugeneruota 1 skaičių: 988
Sugeneruota 2 skaičių: 1046
Sugeneruota 3 skaičių: 966
Sugeneruota 4 skaičių: 969
Sugeneruota 5 skaičių: 1023
Sugeneruota 6 skaičių: 1016
Sugeneruota 7 skaičių: 974
Sugeneruota 8 skaičių: 1021
Sugeneruota 9 skaičių: 1028
Sugeneruota 10 skaičių: 969

Sugeneruota 1 skaičių: 100409
Sugeneruota 2 skaičių: 99689
Sugeneruota 3 skaičių: 99961
Sugeneruota 4 skaičių: 100404
Sugeneruota 5 skaičių: 99452
Sugeneruota 6 skaičių: 100208
Sugeneruota 7 skaičių: 100332
Sugeneruota 8 skaičių: 99902
Sugeneruota 9 skaičių: 99833
Sugeneruota 10 skaičių: 99810

ciklai: Uzduotis8 - “for, for”

Turim skaičių masyvą **pirmas** = {5, 5, 3, 9, 6, 1, 1, 7, 8, 1};. Reikia rasti didžiausią reikšmę

PS. galbut reikalingas pagalbinis kintamasis

ciklai: Uzduotis8a - “for, for”

Turim skaičių masyvą **pirmas** = {5, 5, 3, 9, 6, 1, 1, 7, 8, 1};. Reikia surikiuos skaičius didėjimo tvarką.

PS. galbut reikalingas pagalbinis kintamasis

break ir continue komandos

- Java turi galimybę lanksčiai reguliuoti ciklo kamieno vykdymą.
 - Tarkime jūs turite ciklą kuris paprastai vykdomas 10 kartų, bet ketvirtadieniais jį reikia atlikti tik keturis kartus.
 - Panašaus tipo atvejais naudojami programos vykdymo eigą keičiantys Java operatoriai **break** ir **continue**.

break komanda

- **break** - komanda naudojama perkelti programos vykdymą už ciklo (for, do, while, switch) konstrukcijos/kamino/bloko {} pabaigos.
- Ciklas bus užbaigiamas nepaisant jo palyginimo reikšmės ir bus vykdoma einanti po ciklo pabaigos komanda.
- Užbaigiant while ciklą galime panaudoti break komandą, be kurios ciklas gali ir niekada nesibaigti:
 - `int i=0;`
 - `while(true) {`
 - `System.out.println(i);`
 - `i++;`
 - `if (i > 10)`
 - `break;`
 - `}`

continue komanda

- **continue** komanda panaši į **break** komandą
- **continue** komanda priverčia programos vykdymą tęsti praleidžiant po sutikto **continue** ciklo kamieno komandas
 - Pavyzdžiui pateiktame kode paprastai išvengiama dalyba iš nulio:

```
for(int i = -10; i < 10; i++) {  
    if (i == 0)  
        continue;  
    System.out.println(1/i);  
}
```

Šis ciklas neatliks dalybos, kai $i == 0$. Atkreipkime dėmesį, kad $i++$ bus apskaičiuojamas ir tuo atveju, kai kamieno vykdymą nutrauks **continue** operatorius.

Jei taip nebūtų, tai parašytas kodas būtų vykdomas be galo, nes nepakistų i reikšmė 0.

ciklai: Uzduotis9

Apsirašyti `int[]` skaitiniai masyvą iš 100 elementų. Priskirti elementams atsitiktinai sugeneruota reikšmę nuo 0 iki 1000.

1. Apskaičiuoti mažiausią rastą reikšmę
2. Apskaičiuoti didžiausią rastą reikšmę
3. Apskaičiuoti elementų sumą ir vidurkį
4. Išspausdinti pirmo sutikto masyve elemento indeksą kur reikšmė lygi 500. **break**
5. Apskaičiuoti elementų, kurie didesni už 500 vidurkį (pasinadoti **continue**).



```
import java.util.Random;
```

```
public class HelloWorld{
```

```
    public static void main(String[] args) {
```

```
        int [] masyvas = new int[20];
```

```
        Random random = new Random();
```

```
        for (int indeksas = 0; indeksas < masyvas.lenght; indeksas++){
```

```
            masyvas[indeksas] = random.nextInt(1000);
```

```
        }
```

```
        //1. max
```

```
        int max = masyvas[0];
```

Uzduotis9

```
public static void main(String[] args) {  
    int[] skaicia = new int[100];  
    for (int i = 0; i < skaicia.length; i++) {  
        skaicia[i] = random.nextInt(1000);  
    }  
    int minValue = skaicia[0];  
    int maxValue = skaicia[0];  
    int suma = 0;  
    for (int i = 0; i < skaicia.length; i++) {  
        if (minValue > skaicia[i]) {  
            minValue = skaicia[i];  
        }  
        if (maxValue < skaicia[i]) {  
            maxValue = skaicia[i];  
        }  
        suma += skaicia[i];  
    }  
    int vidurkis = suma / skaicia.length;  
    for (int i = 0; i < skaicia.length; i++) {  
        if (skaicia[i] == 500) {  
            System.out.println("Pirmas indeksas su 500 reikšme: " + i);  
            break;  
        }  
    }  
    int suma500 = 0;  
    for (int i = 0; i < skaicia.length; i++) {  
        if (skaicia[i] < 500) {  
            continue;  
        }  
        suma500 += skaicia[i];  
    }  
}
```

Break ir continue

break – nutraukia ciklą

continue – nutraukia dabartinę iteraciją ir grįžta į ciklo pradžią

****Po šių operatorių gali būti rašoma žymė, tai leidžia iš karto nutraukti kelis įdėtus ciklus. Žymės sintaksė yra tokia: pirma rašomas žymės vardas, po to dvitaškis ir toliau einantis programos kodas. Praktiškai tai atrodo taip:*

```
loop: for(int i=0; i< 10; i++) {  
  
}
```

***Žymės naudojimas

```
1.  label1:
2.  išorinis_ciklas {
3.  vidinis_ciklas {
4.  //...
5.  break; // 1
6.  //...
7.  continue; // 2
8.  //...
9.  continue label1; // 3
10. //...
11. break label1; // 4
12. }
13. }
```

***Žymės PVZ

```
int i = -1, j = -1;
int nums[][] = new int[5][5];
boolean found = false;

loop: for (i = 0; i < 5; i++) {
    for (j = 0; j < 5; j++) {
        if (nums[i][j] == 5) {
            found = true;
            break loop;
        }
    }
}

if (!found)
    System.out.println("Reikšmė nerasta");
else
    System.out.println("Rasta taške " + i + "," + j);
```

return komanda

- **return** komanda panaši į **break**, tik ja yra užbaigiamas ne ciklo bet metodo vykdymas.
- Papildomai po return gali būti nurodyta išraiška, kuri bus apskaičiuota ir grąžinta iškvietusiam metodui.

Pvz2

Main:

```
| while |  
    | for |  
        | do |  
            |return
```

```
public class Pvz2 {  
    public static void main(String[] args) {  
        System.out.println("Startas");  
        System.out.println(metodasSuCiklu());  
        System.out.println("Pabaiga");  
    }  
    private static String metodasSuCiklu() {  
        while (true) {  
            System.out.println("C1");  
            for (int i = 0; i < 10; i++) {  
                System.out.println("C2");  
                do {  
                    System.out.println("C3");  
                    return "exit";  
                } while (i > 9);  
            }  
        }  
    }  
}
```


Pvz2

Main:

| while |
| for |
| do |
| return

Startas
C1
C2
C3
exit
Pabaiga

```
public class Pvz2 {  
    public static void main(String[] args) {  
        System.out.println("Startas");  
        System.out.println(metodasSuCiklu());  
        System.out.println("Pabaiga");  
    }  
    private static String metodasSuCiklu() {  
        while (true) {  
            System.out.println("C1");  
            for (int i = 0; i < 10; i++) {  
                System.out.println("C2");  
                do {  
                    System.out.println("C3");  
                    return "exit";  
                } while (i > 9);  
            }  
        }  
    }  
}
```

Pvz3

... void (....)

....

return; //!!!

Startas
C1
C2
C3
Pabaiga

```
public class Pvz3 {  
    public static void main(String[] args) {  
        System.out.println("Startas");  
        metodasSuCiklu();  
        System.out.println("Pabaiga");  
    }  
  
    private static void metodasSuCiklu() {  
        while (true) {  
            System.out.println("C1");  
            for (int i = 0; i < 10; i++) {  
                System.out.println("C2");  
                do {  
                    System.out.println("C3");  
                    return;  
                } while (i > 9);  
            }  
        }  
    }  
}
```

ciklai: Uzduotis10

Sukurit metodą kuris pakelia skaičių **X** laipsniu **L**.

main metode išbandyti metodą su įvairiais skaičiais. Išspausdinti rezultatu

X^L

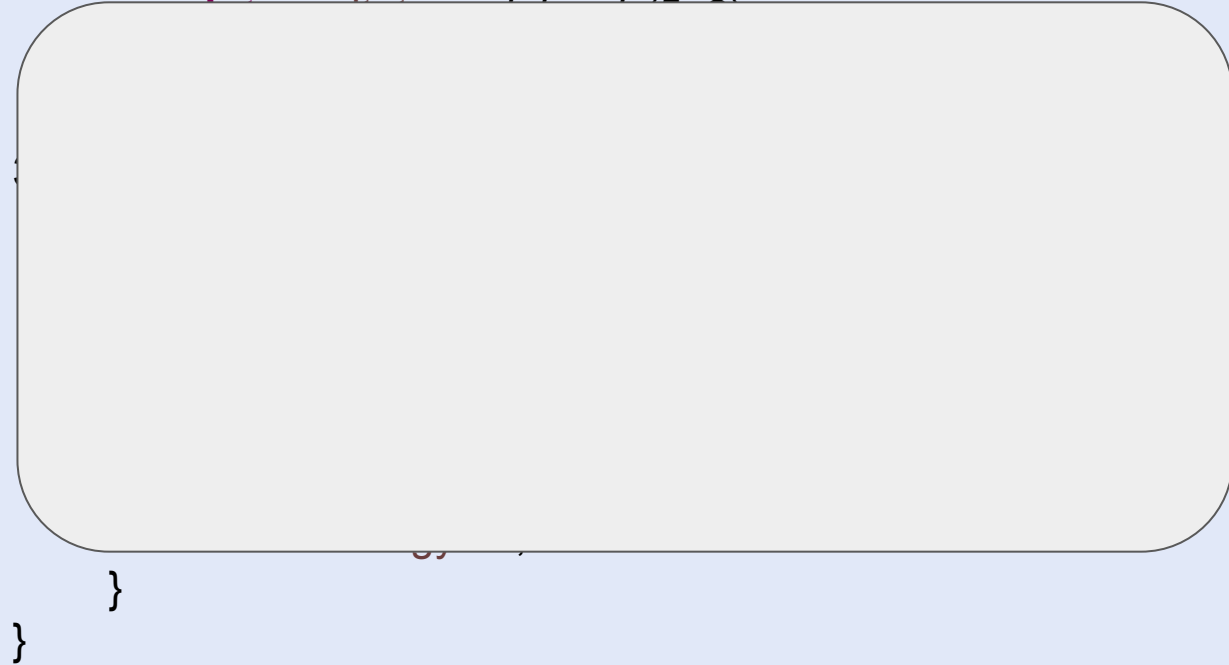
pvz:

$$5^2 = ???;$$

$$256^3 = ???;$$

Uzduotis10

```
public class Uzduotis10 {  
    public static void main(String[] args) {
```



Uzduotis10

```
public class Uzduotis10 {  
    public static void main(String[] args) {
```

```
        3));
```

```
    }
```

```
    }
```

```
        return daugyba;
```

```
    }
```

```
}
```

ciklai: Uzduotis 11

Fibonačio skaičiai

Fibonačio skaičiai tenkina sąlygą:

$$S_n = S_{n-1} + S_{n-2}, \text{ o } S_0 = 1 \text{ ir } S_1 = 1$$

Rasti pirmus 8 Fibonačio skaičius.

Fibonačio skaičių apskaičiavimo logiką vertinga būtų iškelti į naują metodą.

Uzduotis11

```
public class Uzduotis11 {
```

```
    public static void main(String[] args) {
```

```
    }
```

```
    private static int[] fibonacioSk(int dydis) {
```

```
    }
```

```
}
```

Uzduotis11

```
public class Uzduotis11 {  
  
    public static void main(String[] args) {  
        int[] fibonacciSk = fibonacciSk(8);  
        for (int i = 0; i < fibonacciSk.length; i++) {  
            System.out.println(i + " elementas: " + fibonacciSk[i]);  
        }  
    }  
  
    private static int[] fibonacciSk(int dydis) {  
        int[] s = new int[dydis];  
  
        s[0] = 1;  
        s[1] = 1;  
        for (int n = 2; n < dydis; n++) {  
            s[n] = s[n - 1] + s[n - 2];  
        }  
  
        return s;  
    }  
}
```




-
1. Susikurkite klasę, turinčią tik main metodą.
 2. Main metode parašykite programos fragmentą, kuris surastų ir atspausdintų visus pirminius skaičius tarp 1000 ir 1100. Kaip išorinį ciklą naudokite for ciklą.
 3. Pakeiskite programą taip, kad ji surastų pirmuosius 10 pirminių skaičių, didesnių už 1000. Ir toliau naudokite for ciklą.
 4. Pakeiskite išorinį for ciklą ciklu while.
 5. Pakeiskite išorinį ciklą ciklu do-while.
 6. Susikurkite naują klasę, turinčią tik main metodą. Main metode parašykite switch sakinį, kurio kiekviena case šaka spausdina pranešimą. Įdėkite šį switch sakinį į for ciklą taip, kad būtų išbandyta kiekvieną case šaka. Po kiekvienos case šakos įdėkite break operatorių ir išbandykite. Pašalinkite break sakinius ir dar kartą išbandykite. Stebėkite, kaip pasikeitė rezultatas.

Klausimai

?