Materiał pomocniczy do kursu „Podstawy programowania”

Autor: Grzegorz Góralski
ggoralski.com

Pętle

for, while, do ... while, foreach

Jeszcze o operatorach...

- Skrócone operatory arytmetyczne

przykład	znaczenie
x += y	x = x+y
x -= y	x = x-y
x *= y	x = x*y
x /= y	x = x/y

```
int x=8;
System.out.println("Wartość x na początku to "+x);
x += 2;
System.out.println("Po x += 2 wartość x = "+x);
x -= 2;
System.out.println("Po x -= 2 wartość x = "+x);
x *= 2;
System.out.println("Po x *= 2 wartość x = "+x);
x /= 2;
System.out.println("Po x /= 2 wartość x = "+x);
```

- operatory **inkrementacji /dekrementacji** (zwiększania/zmniejszania)

przykład	znaczenie
wartość x pobierana **przed** zmianą wartości	
x++	x = x+1
x--	x = x-1
wartość x pobierana **po** zmianie wartości	
--x	x = x-1
++x	x = x+1

```
int y;
int x=8;
y = 2 * x++;
System.out.println("Po x = 8 i y = 2 * x++ wartość x = "+x+" y = "+y);
x = 8;
y = 2 * ++x;
System.out.println("Po x = 8 i y = 2 * ++x wartość x = "+x+" y = "+y);
x = 8;
System.out.println("Po x = 8 i x++ wartość x = "+ x++);
x = 8;
System.out.println("Po x = 8 i ++x wartość x = "+ ++x);
```

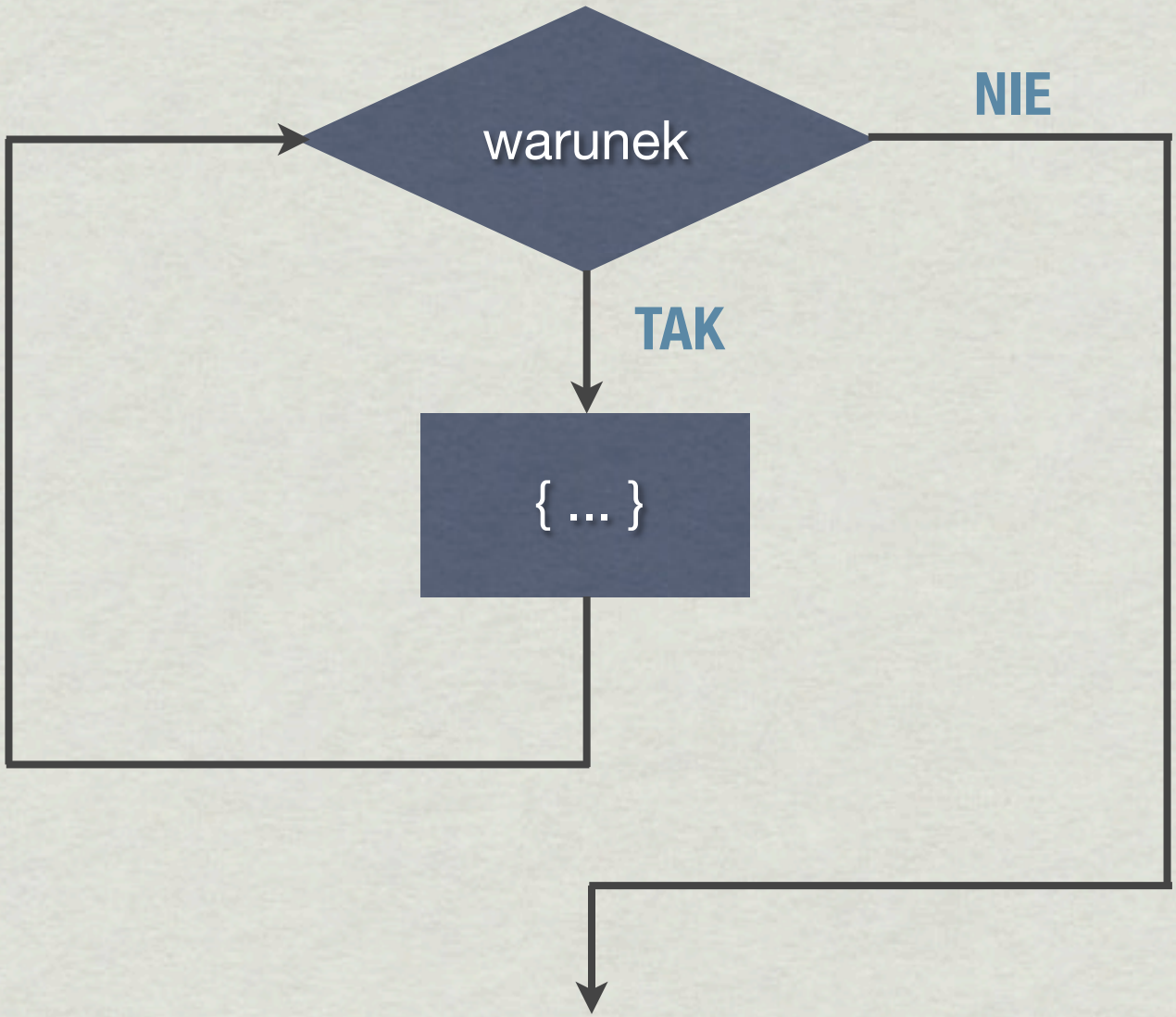
Napisz to za karę 100 razy!

- Często programy muszą wykonywać wielokrotnie tę samą czynność. Na przykład wypisać ten sam tekst/znak.
- Można to zrobic tak:

```
System.out.print("Będę ćwiczyć programowanie w języku Java. ");
System.out.print("Będę ćwiczyć programowanie w języku Java. ");
System.out.print("Będę ćwiczyć programowanie w języku Java. ");
System.out.print("Będę ćwiczyć programowanie w języku Java. ");
.....
```

- Takie rozwiązanie, szczególnie gdy liczba powtórzeń nie jest z góry ustalona, ma wiele wad.
- Na szczęście wymyślono **pętle.**

Ogólna zasada działania pętli



for ... (wersja podstawowa)

(deklaracja i)
inicjalizacja zmiennej

test logiczny zwracający
`true` lub `false`,
dopóki zwraca `true`,
pętla wykonuje się.

zmiana wartości
zmiennej

```
for (inicjalizacja; warunek ; zmiana) {...}
```

Deklaracja zmiennej `i`
której przypisuje się
wartośc 0

polecenia wykonują się,
dopóki zmienna `i` jest
mniejsza niż 10

po każdej **iteracji***,
wartość `i` jest
zwiększana o 1

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

blok poleceń
wykonywanych w pętli

*"wykonaniu poleceń w pętli"

for ...

- Wykonaj kod:

```
for (int i=0; i<100 ; i++) {
    System.out.println(i+
    " Będę w domu ćwiczyć programowanie w języku Java. ");
}
```

- Otrzymujemy:

```
0 Będę w domu ćwiczyć programowanie w języku Java.
1 Będę w domu ćwiczyć programowanie w języku Java.
2 Będę w domu ćwiczyć programowanie w języku Java.
3 Będę w domu ćwiczyć programowanie w języku Java.
...
97 Będę w domu ćwiczyć programowanie w języku Java.
98 Będę w domu ćwiczyć programowanie w języku Java.
99 Będę w domu ćwiczyć programowanie w języku Java.
```

for ...

- Gdybyśmy chcieli otrzymać numerację od 1 do 100 :

```
for (int i=1; i<=100 ; i++) {
    System.out.println(i+
    " Będę w domu ćwiczyć programowanie w języku Java. ");
}
```

- Otrzymujemy:

```
 1 Będę w domu ćwiczyć programowanie w języku Java.
 2 Będę w domu ćwiczyć programowanie w języku Java.
 3 Będę w domu ćwiczyć programowanie w języku Java.
 ...
 98 Będę w domu ćwiczyć programowanie w języku Java.
 99 Będę w domu ćwiczyć programowanie w języku Java.
 100 Będę w domu ćwiczyć programowanie w języku Java.
```

while ...

test logiczny zwracający `true` lub `false`,
dopóki zwraca `true`, pętla wykonuje się.

```
while (warunek) {...}
```

while ...

- Uruchom kod:

```
int i=1;
while (i<=100) {
    System.out.println(i+
    " Będę w domu ćwiczyć programowanie w języku Java. ");
    i++;
}
```

- Otrzymujemy:

```
1 Będę w domu ćwiczyć programowanie w języku Java.
2 Będę w domu ćwiczyć programowanie w języku Java.
3 Będę w domu ćwiczyć programowanie w języku Java.
...
98 Będę w domu ćwiczyć programowanie w języku Java.
99 Będę w domu ćwiczyć programowanie w języku Java.
100 Będę w domu ćwiczyć programowanie w języku Java.
```

while ...

- Można też tak:

```
int i=0;
while (i<100) {
    i++;
    System.out.println(i+
    " Będę w domu ćwiczyć programowanie w języku Java. ");
}
```

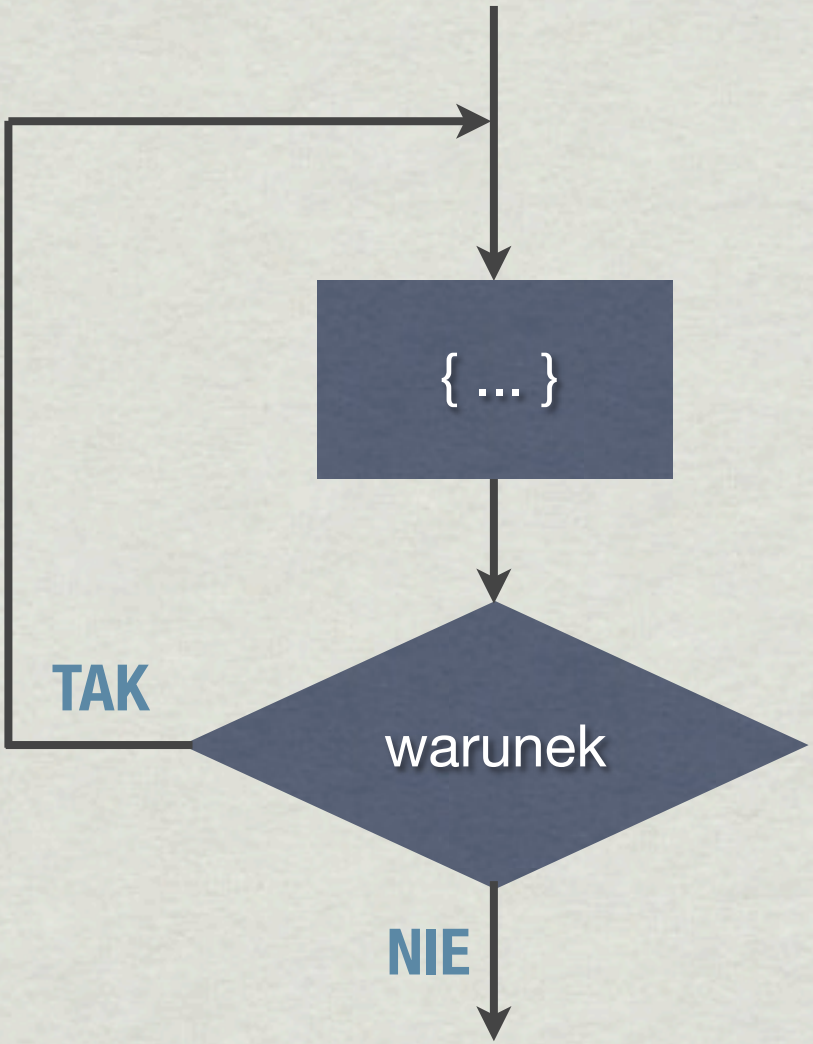
- Otrzymujemy:

```
1 Będę w domu ćwiczyć programowanie w języku Java.
2 Będę w domu ćwiczyć programowanie w języku Java.
3 Będę w domu ćwiczyć programowanie w języku Java.
...
98 Będę w domu ćwiczyć programowanie w języku Java.
99 Będę w domu ćwiczyć programowanie w języku Java.
100 Będę w domu ćwiczyć programowanie w języku Java.
```

do ... while ...

```
do {...} while (warunek)
```

- Ponieważ warunek sprawdzany jest **po** wykonaniu pętli, **najpierw** wykonują się polecenia w niej zawarte
- Konstrukcję tą stosujemy gdy chcemy mieć pewność, że kod w pętli wykona się przynajmniej raz.



while ...

- Uruchom kod:

```
int i=100;
do {
   System.out.println(" i = "+i);
} while (i == 0);
```

- Otrzymujemy:

```
i = 100
```

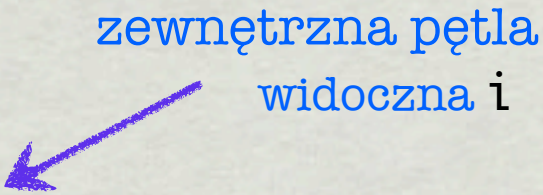
- Kod się wykonał, chociaż warunek nie został spełniony.

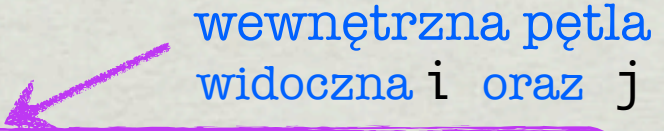
„foreach”

- Ten rodzaj pętli poznamy przy okazji tablic

Zagnieżdżanie pętli

- Pętle można zagnieżdżać, np:

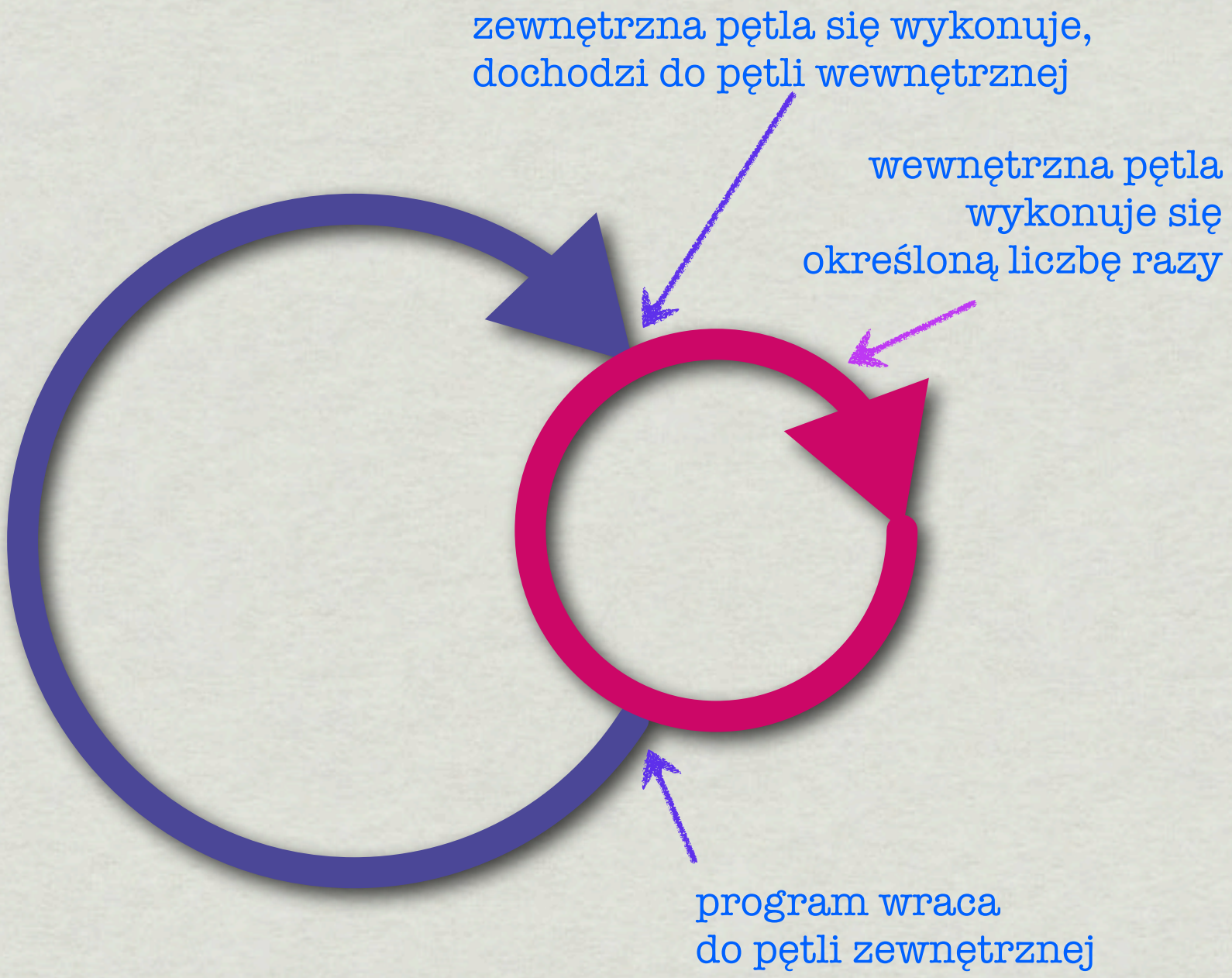





```
for (int i=0; i<5; i++) {
   for (int j=0; j<3; j++ ) {
      System.out.println("i="+i+" j="+j+" : ");
   }
   System.out.println("i="+i);
}
```

UWAGA: Zmienne zadeklarowane w pętli, nie są „widoczne” na zewnątrz, ale można się do nich odwoływać w pętlach wewnętrznych. Dlatego w powyższym przykładzie w pętli wewnętrznej możemy się odwołać do zmiennych `i` oraz `j` ale próba odwołania się do zmiennej `j` w pętli zewnętrznej spowodowałaby błąd. Podobnie jest z innymi zagnieżdzonymi blokami kodu np. przy instrukcjach wyboru.

Zagnieżdżanie pętli



Pętle - przykłady

- Napisz program który pobierze od użytkownika liczbę całkowitą, a następnie wypisze wszystkie liczby nieparzyste od zera do podanej liczby.

```
Scanner skaner = new Scanner(System.in);
System.out.print("Podaj liczbę: ");
// Pobiera liczbę z klawiatury
String liczbaK = skaner.nextLine();
// Parsuje String na int
int liczba = Integer.parseInt(liczbaK);
// pętla która zwiększa wartość zmiennej "i" od 0 do
// wartości równej zmiennej "liczba"
for (int i = 0; i <= liczba; i++) {
    // Sprawdzamy, czy liczba jest nieparzysta
    if (i%2 !=0) {
        System.out.println("Kolejna liczba nieparzysta to: "+i);
    }
}
```

Pętle - przykłady

- Ulepsz program tak, aby nie wykonywał się raz, ale działał tak długo aż użytkownik wprowadzi znak **K** lub **k** (jak koniec)

Jeśli chcemy wydrukować cudzysłów, należy go poprzedzić znakiem ****

```
Scanner skaner = new Scanner(System.in);
String liczbaK = "0";
int liczba;
while (!liczbaK.equalsIgnoreCase("k")){
    System.out.print("Podaj liczbę, jeśli chcesz wyjść, wprowadź \"k\": ");
    liczbaK = skaner.nextLine();
    // Sprawdzamy, czy użytkownik nie wcisnął "k"
    if (!liczbaK.equalsIgnoreCase("k")){
        liczba = Integer.parseInt(liczbaK);

        for (int i = 0; i <= liczba; i++) {
            // Sprawdzamy, czy liczba jest nieparzysta
            if (i%2 !=0) {
                System.out.println("Kolejna liczba nieparzysta to: "+i);
            }
        }
    }
}
System.out.println("Dziękuję za owocną współpracę.");
```

Pętle - przykłady

- Druga wersja powyższego, ale tym razem umieszczamy kod pobierający informację od użytkownika **pod** pętlą iterującą, która po raz pierwszy się w ogóle nie wykonuje. Dzięki temu unikamy powtórnego sprawdzania wartości wprowadzonego „Stringa”.

```
Scanner skaner = new Scanner(System.in);
String liczbaK = "0";
int liczba;
while (!liczbaK.equalsIgnoreCase("k")){
    liczba = Integer.parseInt(liczbaK);
    // pierwszy raz komunikat się nie drukuje, ponieważ liczba == 0 i i%2 ==0.
    for (int i = 0; i <= liczba; i++) {
        if (i%2 !=0) {
            System.out.println("Kolejna liczba nieparzysta to: "+i);
        }
    }
    System.out.print("Podaj liczbę, jeśli chcesz wyjść, wprowadź \"k\": ");
    liczbaK = skaner.nextLine();
}
System.out.println("Dziękuję za owocną współpracę.");
```