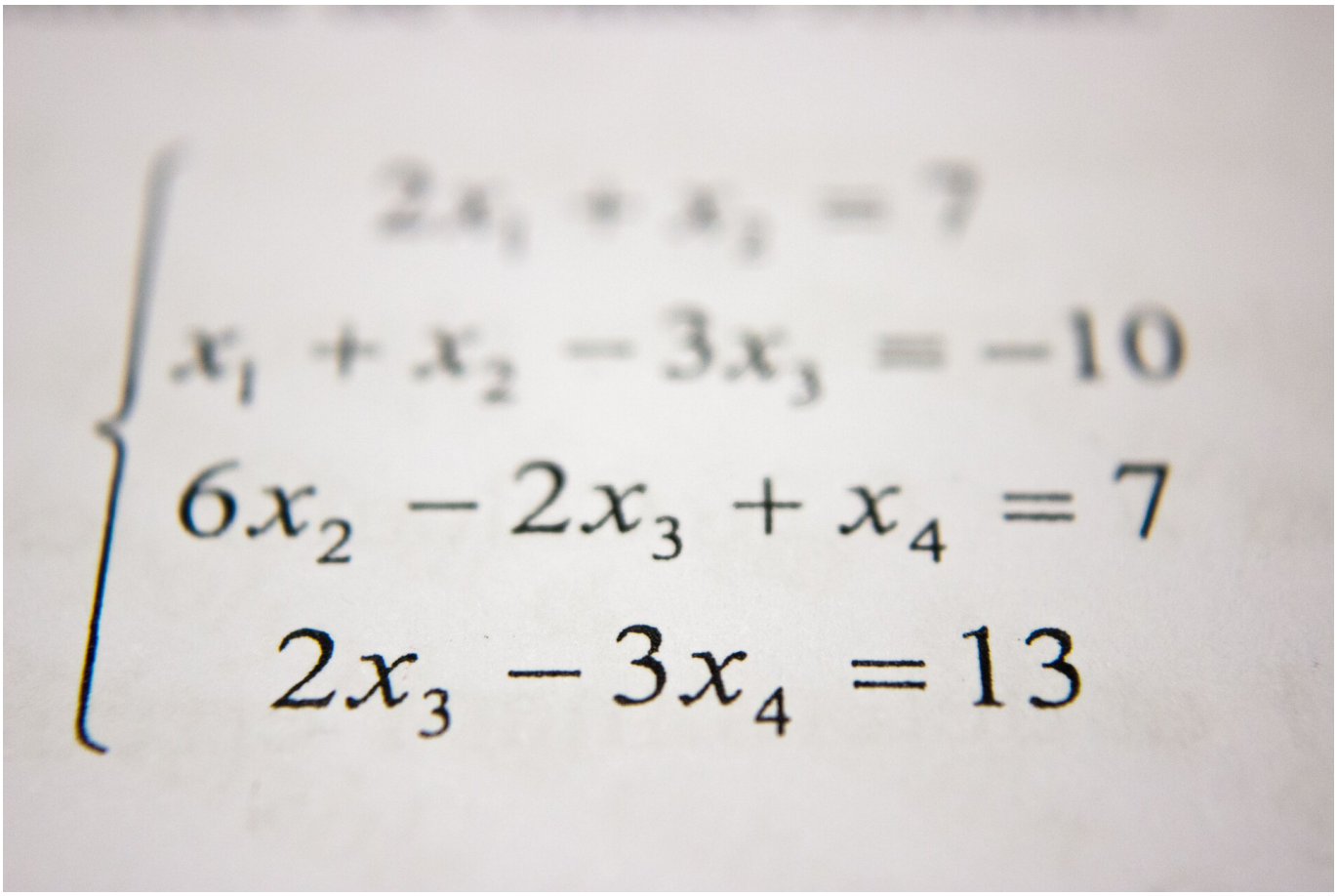
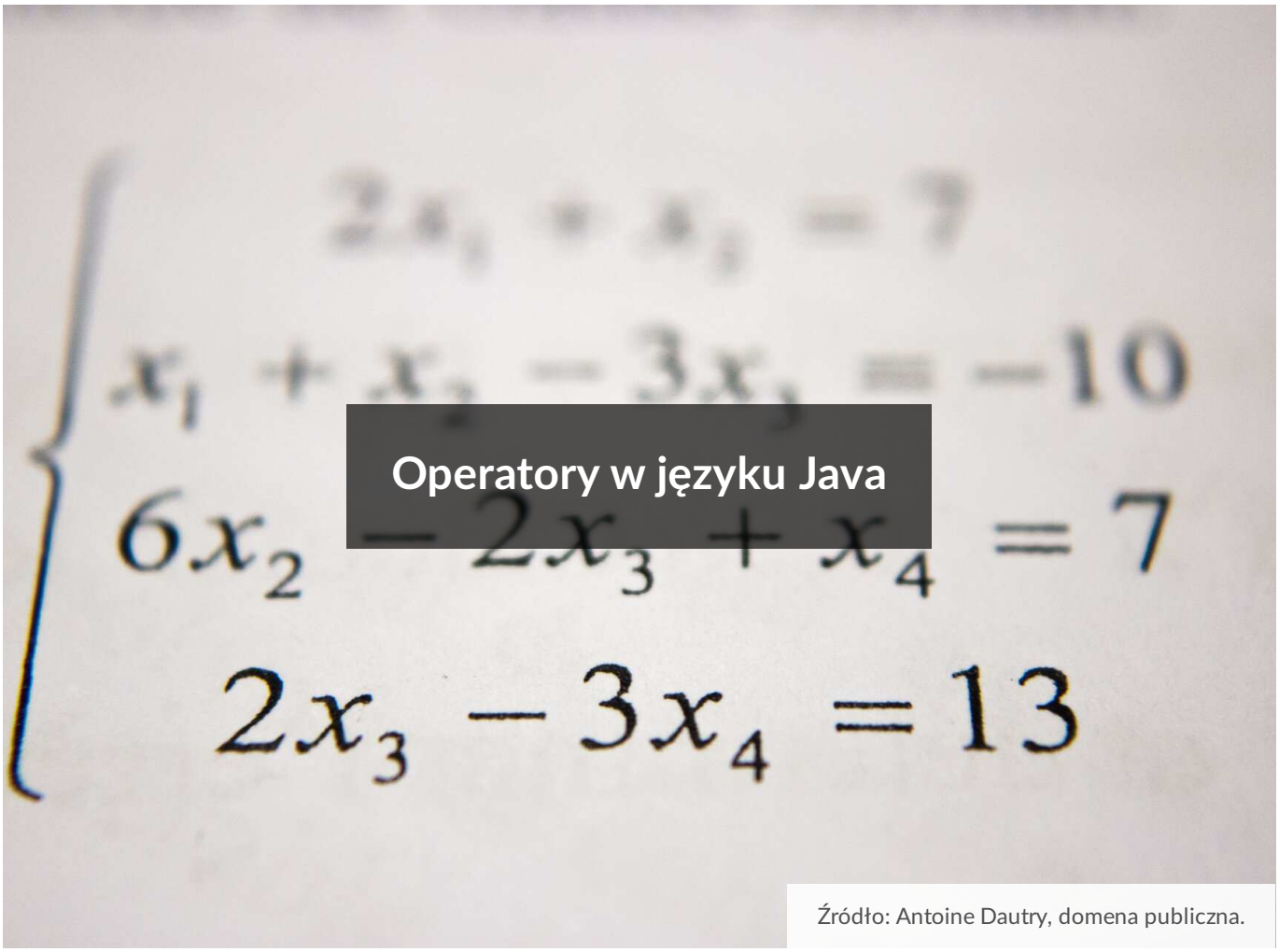


Operatory w języku Java

- Wprowadzenie
- Animacja
- Przeczytaj
- Sprawdź się
- Dla nauczyciela



Źródło: Antoine Dautry, domena publiczna.

Operatory to konstrukcje językowe zwracające pewne wartości. Poznaliśmy już kilka typów operatorów (np. arytmetyczne, logiczne, przypisań); wiemy również, czym się charakteryzują. Podstawowe cechy operatorów to m. in. liczba oraz typ argumentów, typ wyniku, wykonywane działanie, priorytet.

Pozostałe podstawowe informacje na ich temat znajdziesz w e-materiale Operatory.

W tym e-materiale zapoznamy się z operatorami w języku Java.

Charakterystykę operatorów w innych językach programowania znajdziesz w e-materiałach:

- Operatory w języku C++,
- Operatory w języku Python.

Więcej zadań? Sięgnij do Operatory – ćwiczenia.

Twoje cele

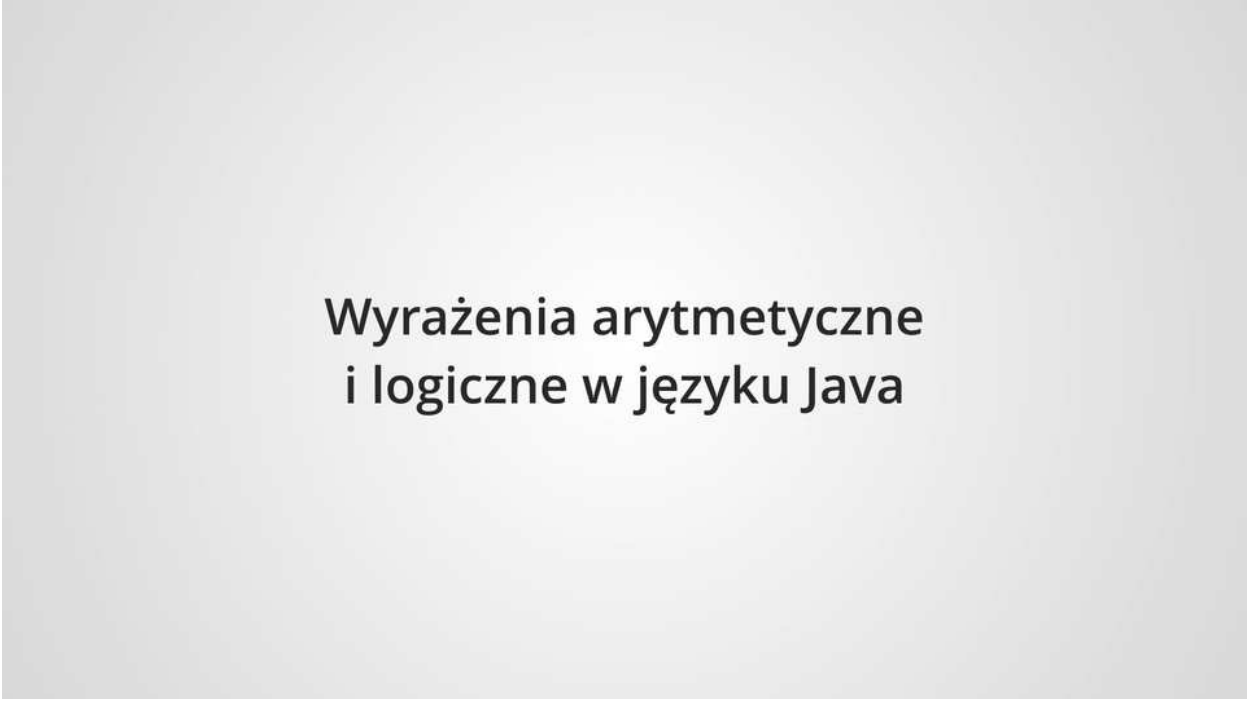
- Scharakteryzujesz operatory matematyczne i logiczne w języku Java.
- Wyjaśnisz, jakie reguły rządzą wyrażeniami w języku Java.
- Przygotujesz proste programy z wykorzystaniem poznanych operatorów.

- Napiszesz proste metody zwracające wynik wyrażenia.

Animacja

Polecenie 1

Zapoznaj się z animacją. Poszukaj informacji na temat innych operatorów. Rozstrzygnij, czy możemy dodawać do siebie zmienne typu `String`.



Film dostępny pod adresem /preview/resource/R1DkZ65ZXuejm

Źródło: Contentplus.pl Sp. z o.o., licencja: CC BY-SA 3.0.

Nagranie filmowe dotyczące wyrażeń arytmetycznych i logicznych w języku Java

Polecenie 2

Napisz prosty kalkulator. Do zmiennych `a` i `b` przypisz wartości całkowite, a następnie wykonaj kolejno działania:

- dodawania,
- odejmowania,
- mnożenia,
- dzielenia,
- dzielenia z resztą.

Specyfikacja problemu:

Dane:

`a`, `b` – liczby całkowite; `b` $\neq$ `0`.

Wynik:

Program wypisuje w kolejnych wierszach działania (wraz z wynikami) w takiej kolejności, jak podano w poleceniu.

Przykładowe wyjście:

```
a + b
9
a - b
- 1
a * b
20
a / b
0
a % b
```

```
4
```

```
public class MyClass {
    public static void main(String[] args){
        // Tutaj dodaj własny kod. Użyj funkcji
        // System.out.print();
    }
}
```

```
```

Polecenie 3

Latami przestępnymi nazywamy te, których numeracja:

- jest podzielna przez 4 oraz jest niepodzielna przez 100;
- jest podzielna przez 400.

Napisz program sprawdzający, czy dany rok jest przestępny.

Specyfikacja problemu:

Dane:

- `rok` – zmienna przechowująca badany rok; liczba naturalna

Wynik:

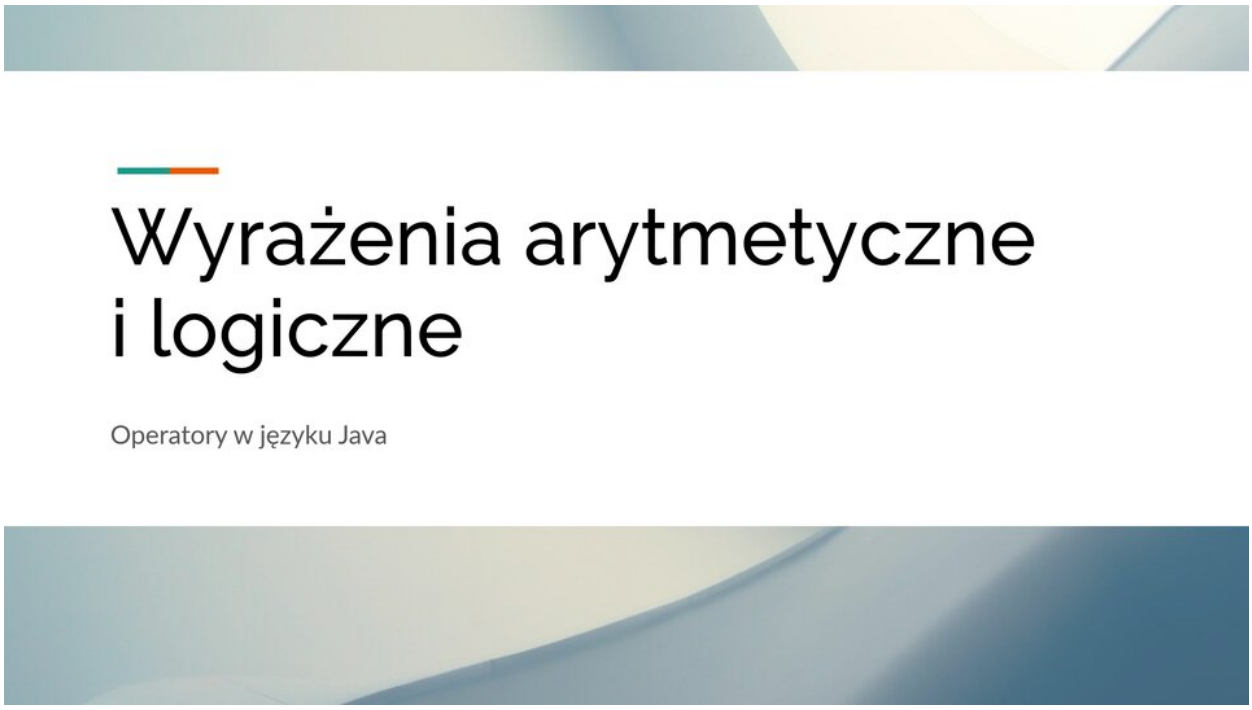
Program wypisuje wartość 1, gdy zmienna `rok` przechowuje informacje o roku przestępnym i 0 w przeciwnym przypadku.

```
public class MyClass {
    public static void main(String[] args){
        // Tutaj dodaj własny kod. Użyj funkcji
        // System.out.print();
    }
}
```

```
1
```

Polecenie 4

Zapoznaj się z filmem i porównaj z nim swoje rozwiązania.



Film dostępny pod adresem /preview/resource/R1GXLFdZ8uE27

Źródło: Contentplus.pl Sp. z o.o., licencja: CC BY-SA 3.0.

Nagranie filmowe dotyczące wyrażeń arytmetycznych i logicznych, operatorów w Javie

Przeczytaj

Podsumowanie

Wyrażenia mogą się składać ze zmiennych, wartości i operatorów. Analogicznie do wyrażenia matematycznego:

```
x + 4 = 6
```

w języku Java mamy wyrażenie:

```
int jednaCzwarta = x / 4;
```

gdzie wynik `jednaCzwarta` jest zmienną, `x` jest zmienną, $/$ jest operatorem (dzielenia), a 4 jest konkretną wartością, czyli stałą.

Wyrażenia arytmetyczne

Tak w matematyce, jak i w języku Java można wykonywać działania arytmetyczne. Służą do tego operatory:

- dodawania $+$,
- odejmowania $-$,
- mnożenia $*$,
- dzielenia $/$,
- reszty z dzielenia (modulo) $\%$.

Żeby zaprezentować działanie operatorów arytmetycznych, stworzymy dwie zmienne. Następnie wykonamy na nich działania z wykorzystaniem właśnie tych operatorów, a wynik wypiszemy do konsoli:

```
int x = 5;
int y = 6;

int suma = x + y;
int roznica = x - y;
int iloczyn = x * y;
int iloraz = x / y;
```

```
8  int modulo = x % y;
9
10 System.out.println(suma);        // Wynik: 11
11 System.out.println(roznica);     // Wynik: -1
12 System.out.println(iloczyn);     // Wynik: 30
13 System.out.println(iloraz);      // Wynik: 0
14 System.out.println(modulo);      // Wynik: 5
```

Pierwsze trzy wyniki są oczywiste. Dlaczego jednak wynik ilorazu to 0, a wynik operacji modulo to 5?

Wynik ilorazu to 0, ponieważ nadaliśmy mu typ `int`. Zatem część ułamkowa została pominięta. Gdybyśmy chcieli otrzymać dokładny wynik w postaci liczby zmiennoprzecinkowej, wynik oraz co najmniej jedna ze zmiennych w wyrażeniu powinna być w takiej właśnie postaci, np.

```
1 int x = 5;
2 double y = 6.0d;
3 double iloraz = x / y;
```

> **Ważne!**
>
> Modulo to operacja wyznaczania reszty z dzielenia dwóch liczb. W matematyce operacja ta zapisywana jest w postaci wzoru:
>
> $$a \bmod d = r.$$
>
> Resztę z dzielenia możemy wyliczyć za pomocą równania:
>
> $$r = a - d \cdot \left\lfloor \frac{a}{d} \right\rfloor.$$
>
> Gdzie a to dzielna, d dzielnik, $\lfloor x \rfloor$ oznacza podłogę liczby x.

Istnieje także jednoargumentowy operator arytmetyczny $-$.
Zmienia on znak argumentu, czyli wyznacza liczbę przeciwną.

```
1 int liczbaDodatnia = 7;
2 int liczbaPrzeciwna = -liczbaDodatnia;
3
```

```
System.out.println(liczbaPrzeciwna);
// program wyświetli -7
```

Operatory przypisania

Oprócz podstawowego operatora, wyrażanego za pomocą znaku równości, w języku Java istnieją także inne operatory przypisania. Są one stosowane, gdy wykonujemy działania na dwóch zmiennych i chcemy przypisać wynik do jednej z nich.

Stosuje się je najczęściej w formie skróconej:

```
int x = 5;
int y = 6;

x += y; // x = x + y;
x -= y; // x = x - y;
x *= y; // x = x * y;
x /= y; // x = x / y;
x %= y; // x = x % y;
```

> **Ważne!**
>
> Wyrażenia arytmetyczne są przetwarzane od lewej strony do prawej, **zgodnie z kolejnością wykonywania działań w matematyce**.
>
> I tak jak w matematyce, możemy tę kolejność zmienić, stosując nawiasy.

Operatory relacyjne

Kolejna grupa operatorów to operatory relacyjne, które porównują ze sobą wartości. Są nimi:

- operator równości ==,
- operator „różny od” ! =,
- operatory silnych nierówności > i <,
- operatory słabych nierówności >= i <=.

Operatory porównania zwracają wartości logiczne `true` lub `false`.

```
double liczba1 = 4 / 5;
double liczba2 = 12.0d;

```

```
boolean rowny = liczba1 == liczba2;
boolean rozny = liczba1 != liczba2;
boolean wieksze = liczba1 > liczba2;

System.out.println(rowny);      // Wynik: false
System.out.println(rozny);      // Wynik: true
System.out.println(wieksze);    // Wynik: false
```

Zmiennej typu boolean możemy przypisać także całe działanie i zwróci prawdę lub fałsz, np.

```
boolean czyWynikTo5 = 2 / 4 * 4 == 5;
System.out.println(czyWynikTo5);    // Wynik: false
```

> **Ważne!**
>
> **Operatora porównania** == nie należy mylić z **operatorem przypisania** =. Choć podobne, znaczą coś innego. Pierwszy z nich sprawdza, czy wartości są równe, natomiast drugi przypisuje lub podstawia wartości do zmiennych.

Operatory logiczne

W języku Java istnieją również trzy podstawowe operatory logiczne:

- && to koniunkcja, czyli spójnik „i" (AND),
- || to alternatywa, czyli spójnik „lub" (OR),
- ! to operator negacji, odwraca on wartość logiczną argumentu (NOT).

Operatory logiczne to spójniki koniunkcji, alternatywy oraz negacji w logice. Stosuje się je do łączenia dwóch lub więcej wartości (zmiennych), wyrażeń lub warunków.

Operatory logiczne również zwracają wartości logiczne:

- koniunkcja zwróci `true` wtedy i tylko wtedy, gdy wszystkie elementy są prawdą,
- alternatywa zwróci `true`, gdy przynajmniej jeden z elementów jest prawdą,
- negacja zwróci `true`, gdy argument ma wartość `false`, a `false`, gdy argument ma wartość `true`.

Operatorów logicznych używa się m.in. w połączeniu z operatorami porównania do tworzenia instrukcji wyboru.

```
double x = 9.8d;
```

```
2 double y = 5.7d;
3
4 boolean czyWieksze = x > y;
5 boolean czyRowne = x == y;
6
7 boolean koniunkcja = czyWieksze && czyRowne;
8 boolean alternatywa == czyWieksze || czyRowne;
9
10 System.out.println(koniunkcja);     // Wynik: false
11 System.out.println(alternatywa);    // Wynik: true
```

Dla powyższych danych `czyWieksze` jest prawdą, a `czyRowne` jest fałszem. Stąd koniunkcja zwróci `false`, ponieważ `true && false == false`. Natomiast alternatywa zwróci `true`, ponieważ wystarczy, że jeden warunek (w tym przypadku `czyWieksze`) jest prawdą.

Istnieje jeszcze jeden specyficzny operator: różnica symetryczna XOR *(eXclusive OR)*. Jest on zapisywany za pomocą symbolu `^`.

Zwraca on wartość `true` wtedy i tylko wtedy, gdy jeden (i nie więcej) z argumentów jest prawdziwy, dla operacji dwuargumentowych. Dla wyrażeń mających więcej argumentów zwraca `true`, gdy liczba argumentów prawdziwych jest nieparzysta.

Operatory inkrementacji i dekrementacji

Kolejną grupą operatorów w języku Java są operatory inkrementacji i dekrementacji, które można zaliczyć do operatorów arytmetycznych. Powodują one zwiększenie (inkrementację) lub zmniejszenie (dekrementację) zmiennej o jeden. Stosuje się je najczęściej w pętlach do zmiany wartości licznika. Operatory te dzielą się na:

- postfiksowe; `x++` oraz `x--`, w których wartość `x` jest zwiększana lub zmniejszana **po** przypisaniu do zmiennej,
- prefiksowe; `++x` oraz `--x`, w których wartość `x` jest zwiększana lub zmniejszana **przed** przypisaniem do zmiennej.

Możemy więc także zapisać działanie `x++` równoważnie jako `x = x + 1` lub `x += 1`.

Kolejność wykonywania działań

W języku Java działania wykonywane są, podobnie jak w matematyce, według ustalonych priorytetów, a gdy te są równe, wyrażenie przetwarzane jest od lewej do prawej. Kolejność tę możemy zmieniać, korzystając z nawiasów.

Listę priorytetów można przedstawić za pomocą tabeli:

Priorytet	Operatory	Opis
1	++x, --x, !x, -x	operatory jednoargumentowe: prefiksowe, negacji i zmiany znaku
2	*, /, %	mnożenie, dzielenie, dzielenie modulo
3	+, -	dodawanie, odejmowanie
4	x++, x--	operatory postfiksowe
5	>, <, >=, \lt=	operatory relacyjne: nierówności
6	==, ! =	operatory relacyjne: równy, różny
7	^	różnica symetryczna
8	////	koniunkcja
9	||	alternatywa
10	=, + =, - =, * =, / =, % =	operatory przypisania

Dla zainteresowanych

W języku Java istnieją jeszcze inne rodzaje operatorów. Są nimi operatory bitowe i trójargumentowy operator związany z instrukcjami warunkowymi.

Słownik

podłoga (część całkowita, cecha)

największa liczba całkowita nie większa od zadanej liczby

pętla

jedna z podstawowych składowych niemalże każdego języka programowania; pętle to instrukcje pozwalające na wykonywanie pewnego fragmentu kodu wielokrotnie

Sprawdź się

Pokaż ćwiczenia:

Ćwiczenie 1

Uporządkuj operatory według priorytetu: od najniższego do najwyższego.

- ||
- ==
- <=
- &&

Ćwiczenie 2

Wskaż operatory przypisania.

- [ ] +=
- [ ] =
- [ ] >=
- [ ] ==

Ćwiczenie 3

Uzupełnij tekst.

Gdy używamy mieszanych typów _____ w języku Java, następuje _____. Może ono być automatyczne lub _____.
Konwersja _____ w Javie następuje automatycznie, natomiast _____ to ta zdefiniowana explicite przez _____.

liczbowych | programistę | danych | zawężająca | Wirtualną Maszynę Javy | jawne | kompilator | rzutowanie | rozszerzająca | zmiennych

Ćwiczenie 4

Połącz w pary.

`%`	operator preinkrementacji
`!`	negacja
`++i`	dzielenie całkowite
`/`	koniunkcja
`x++`	operator postinkrementacji
`&&`	różny od
`!=`	dzielenie modulo

Ćwiczenie 5

Wskaż, jaka wartość zostanie przypisana do zmiennej `zaprzeczenie`.

```
int liczba = 6;
int liczbaUjemna = -(liczba);

boolean ktoraWieksza = liczba < -(liczba);
boolean zaprzeczenie = !(ktoraWieksza);
```

- ◯ `true`
- ◯ `false`

Ćwiczenie 6

Wskaż, jaka wartość zostanie przypisana do zmiennej zaprzeczenie2.

```
int liczba = 6;
int liczbaUjemna = -(liczba);

boolean ktoraWieksza2 = liczba < -(liczba) || liczba >= -(liczba);
boolean zaprzeczenie2 = !(ktoraWieksza2);
```

- ◯ `true`
- ◯ `false`

Ćwiczenie 7

Wskaż, jakie wartości zostaną przypisane do kolejnych zmiennych typu boolean.

```
boolean czyParzysta = 5 % 2 == 0;

double duzaLiczba = 123_454_13232_2313d;
boolean czyDuzaLiczbaWielokrotnosciaTrojki = duzaLiczba % 3 ==
0;

boolean czyTuJestPrawda = czyParzysta ||
czyDuzaLiczbaWielokrotnosciaTrojki;
```

- ○
```
false
false
true
```

- ○
```
false
true
false
```

- ○
```
false
true
true
```

- ○
```
false
false
false
```

Ćwiczenie 8

Wskaż prawdziwe stwierdzenie.

```
boolean czyParzysta = 5 % 2 == 0;
double duzaLiczba = 123_454_13232_2313d;
boolean czyDuzaLiczbaWielokrotnosciaTrojki = duzaLiczba % 3 ==
0;

boolean czyTuJestPrawda = czyParzysta ||
czyDuzaLiczbaWielokrotnosciaTrojki;
boolean koniunkcja = czyDuzaLiczbaWielokrotnosciaTrojki &&
czyTuJestPrawda;

boolean alternatywnie = koniunkcja && czyParzysta;
boolean wyrazenie = alternatywnie ^ koniunkcja ||
czyTuJestPrawda;
```

- ○ `// czyTuJestPrawda zwraca false, a wyrazenie true`
- ○ `// koniunkcja zwraca true, a alternatywnie false`
- ○ `// czyTuJestPrawda i alternatywnie zwracają true`
- ○ `// wyrazenie i alternatywnie zwracają false`

Dla nauczyciela

Autor: Zespół autorski Contentplus.pl sp. z o.o.
Przedmiot: Informatyka

Temat: Operatory w języku Java

Grupa docelowa:

Liceum ogólnokształcące i technikum, liceum ogólnokształcące, technikum, zakres podstawowy i rozszerzony

Podstawa programowa:

Cele kształcenia – wymagania ogólne

> II. Programowanie i rozwiązywanie problemów z wykorzystaniem komputera oraz innych urządzeń cyfrowych: układanie i programowanie algorytmów, organizowanie, wyszukiwanie i udostępnianie informacji, posługiwanie się aplikacjami komputerowymi.

Treści nauczania – wymagania szczegółowe

> II. Programowanie i rozwiązywanie problemów z wykorzystaniem komputera i innych urządzeń cyfrowych.
>
> > Zakres podstawowy. Uczeń:
> >
> > > 2) do realizacji rozwiązań problemów prawidłowo dobiera środowiska informatyczne, aplikacje oraz zasoby, wykorzystuje również elementy robotyki;
> >
> > Zakres rozszerzony. Uczeń spełnia wymagania określone dla zakresu podstawowego, a ponadto:
> >
> > > 3) sprawnie posługuje się zintegrowanym środowiskiem programistycznym przy pisaniu, uruchamianiu i testowaniu programów;

Kształtowane kompetencje kluczowe:

- kompetencje cyfrowe;
- kompetencje osobiste, społeczne i w zakresie umiejętności uczenia się;
- kompetencje matematyczne oraz kompetencje w zakresie nauk przyrodniczych, technologii i inżynierii.

Cele operacyjne (językiem ucznia):

- Scharakteryzujesz operatory matematyczne i logiczne w języku Java.
- Wyjaśnisz, jakie reguły rządzą wyrażeniami w języku Java.
- Przygotujesz proste programy z wykorzystaniem poznanych operatorów.
- Napiszesz proste metody zwracające wynik wyrażenia.

Strategie nauczania:

- konstruktywizm;
- konektywizm.

Metody i techniki nauczania:

- dyskusja;
- rozmowa nauczająca z wykorzystaniem multimedium i ćwiczeń interaktywnych;
- ćwiczenia praktyczne.

Formy pracy:

- praca indywidualna;
- praca w parach;
- praca w grupach;
- praca całego zespołu klasowego.

Środki dydaktyczne:

- komputery z głośnikami, słuchawkami i dostępem do internetu;
- zasoby multimedialne zawarte w e-materiale;
- tablica interaktywna/tablica, pisak/kreda.

Przebieg lekcji

Przed lekcją:

1. **Przygotowanie do zajęć.** Nauczyciel loguje się na platformie i udostępnia e-materiał: „Operatory w języku Java". Uczniowie mają zapoznać się z treściami w sekcji „Przeczytaj".

Faza wstępna:

1. Nauczyciel prosi wybraną osobę o odczytanie tematu lekcji, a następnie określa cele i kryteria sukcesu.

Faza realizacyjna:

1. **Praca z tekstem.** Nauczyciel ocenia, na podstawie informacji na platformie, stan przygotowania uczniów do zajęć. Jeżeli jest ono niewystarczające, prosi wybraną osobę o przedstawienie najważniejszych informacji z sekcji „Przeczytaj".

2. **Praca z multimedium.** Uczniowie w parach realizują polecenie 2 i 3 z sekcji „Animacja”, następnie porównują swoje rozwiązania z przedstawionym na filmie.
3. **Ćwiczenie umiejętności.** Liga zadaniowa - uczniowie wykonują indywidualnie na czas ćwiczenia nr 1-7 z sekcji „Sprawdź się”, a następnie omawiają zadania na forum.

Faza podsumowująca:

1. Nauczyciel ponownie wyświetla na tablicy temat i cele lekcji zawarte w sekcji „Wprowadzenie”. W kontekście ich realizacji następuje omówienie ewentualnych problemów z rozwiązaniem ćwiczeń z sekcji „Sprawdź się”.
2. Wybrany uczeń podsumowuje zajęcia, zwracając uwagę na nabyte umiejętności.

Praca domowa:

1. Uczniowie wykonują ćwiczenie 8 z sekcji „Sprawdź się”.
2. Uczniowie wykonują polecenie 1 z sekcji „Animacja”.

Materiały pomocnicze:

- Oficjalna dokumentacja techniczna dla języka Java SE 8 (lub nowszej wersji).
- Oficjalna dokumentacja techniczna dla oprogramowania Eclipse 4.4 (lub nowszej wersji).

Wskazówki metodyczne:

- Treści w sekcji „Przeczytaj” można wykorzystać na lekcji jako podsumowanie i utrwalenie wiedzy uczniów.