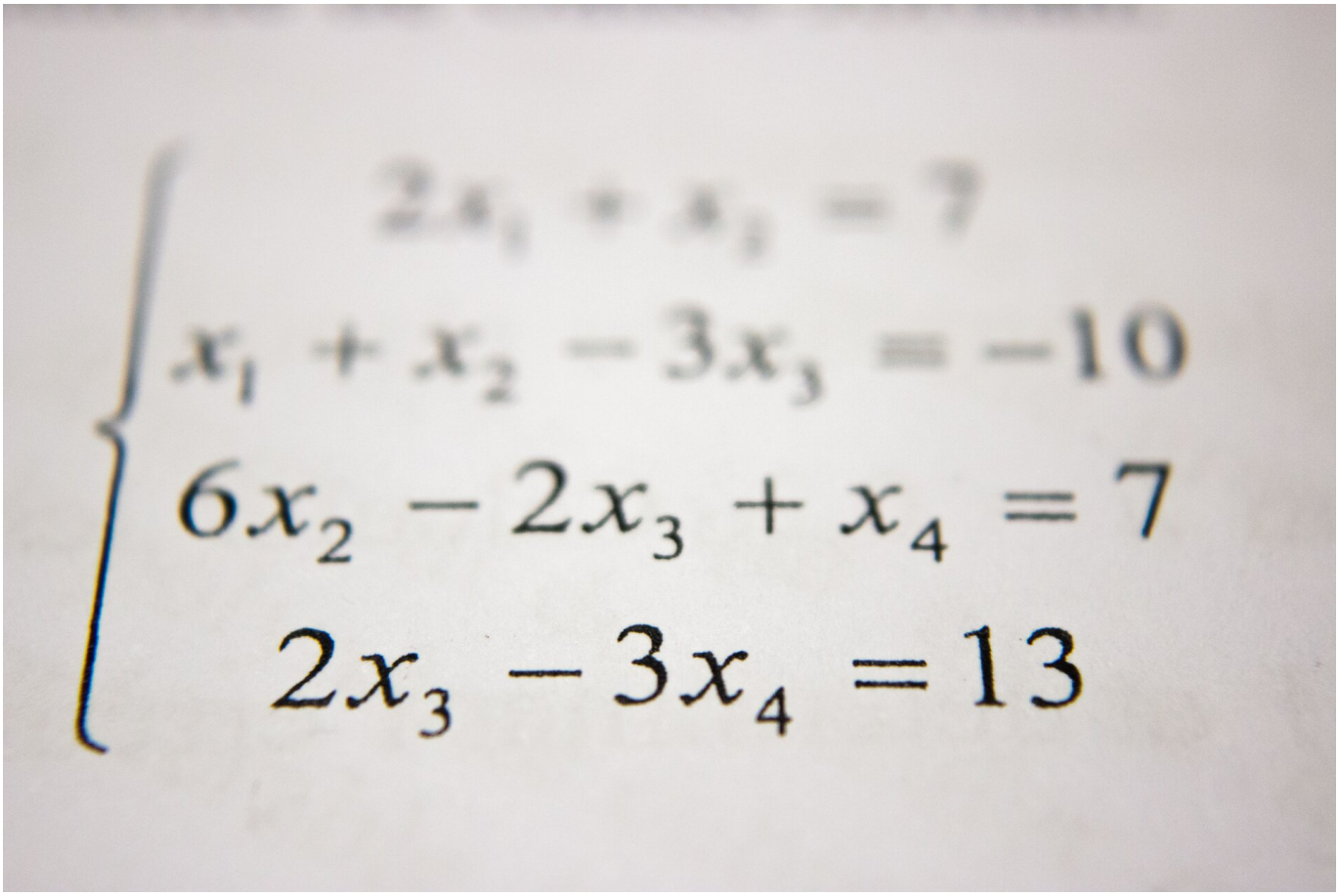
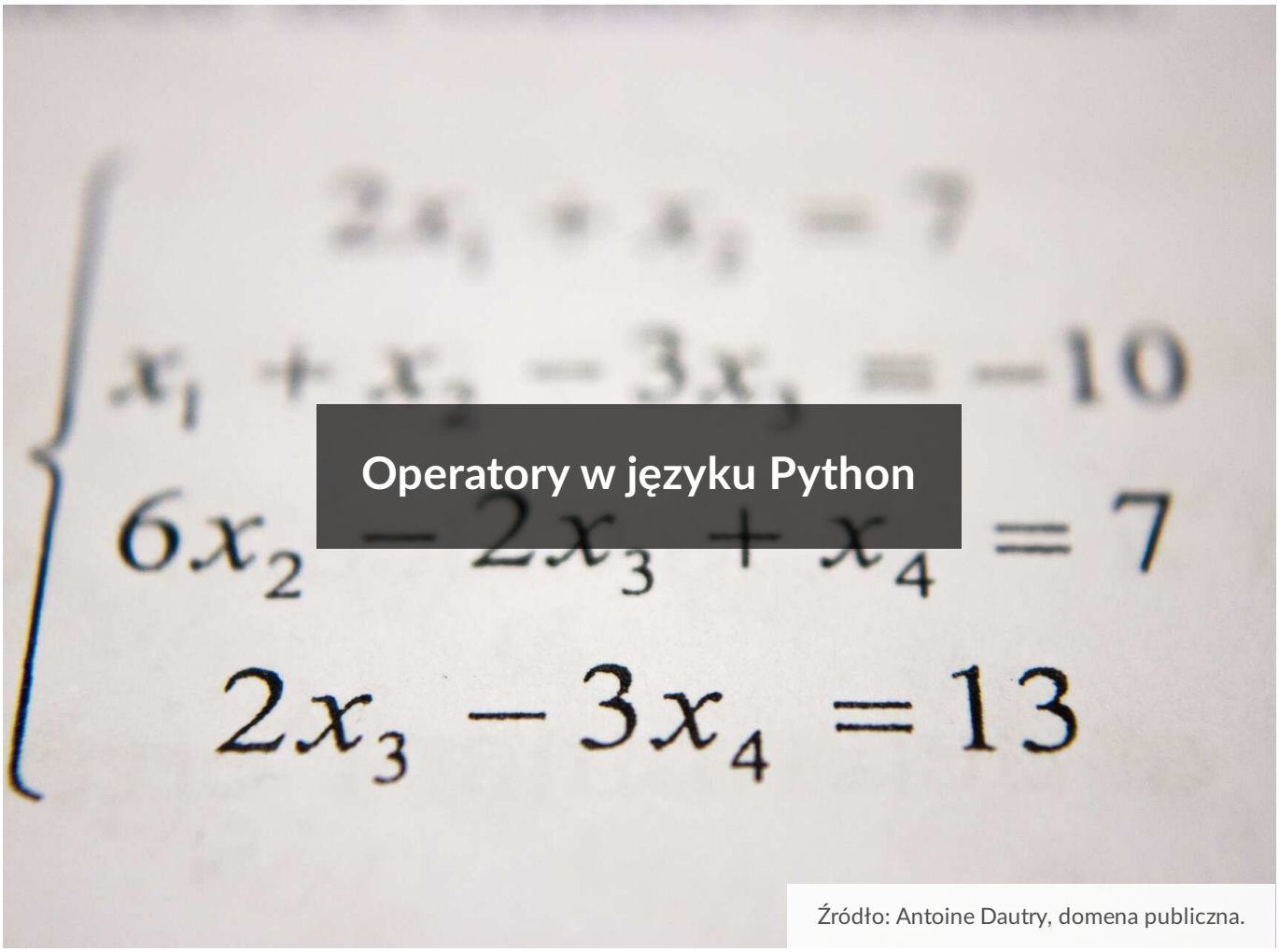


Operatory w języku Python



Źródło: Antoine Dautry, domena publiczna.

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

Podstawowe informacje na ten temat znajdziesz w e-materiale Operatory.

Tym razem zapoznamy się z operatorami w języku Python.

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

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

Więcej zadań? Sięgnij do e-materiału: Operatory – ćwiczenia.

> **Twoje cele**
>
> - Zbudujesz wyrażenia w języku programowania.
> - Scharakteryzujesz operatory arytmetyczne i logiczne oraz ich priorytety w języku Python.

- Przećwiczysz obliczanie wartości wyrażeń.
- Przygotujesz program z wykorzystaniem modułów języka Python.

Schemat interaktywny

Polecenie 1

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

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

Działanie swojego programu przetestuj dla wartości a równej 4 oraz b równej 5.

Specyfikacja problemu:

Dane:

- a – liczba całkowita
- b – liczba całkowita różna od 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
```

```
if __name__ == '__main__':

# Tutaj dodaj własny kod. Użyj funkcji
# print()
```

```
1
```

Polecenie 2

Latami przestępnymi nazywamy te, których numeracja:

- jest podzielna przez 4 oraz jest niepodzielna przez 100 lub:
- jest podzielna przez 400.

Napisz program sprawdzający, czy dany rok jest przestępny.

Specyfikacja problemu:

Dane:

- `rok` – dodatnia liczba naturalna przechowująca badany rok

Wynik:

Jeśli badany rok jest przestępny, program wypisuje wartość `True`; w przeciwnym wypadku – `False`.

Aby powtórzyć wiadomości ze szkoły podstawowej, stwórz program, korzystając ze schematu blokowego.

```
if __name__ == '__main__':
        rok = 2746
# Tutaj dodaj własny kod. Użyj funkcji
# print()
```

```
1
```

Polecenie 3

Zapoznaj się z filmem i porównaj z nim swoje rozwiązanie.

Kod programu (kalkulator) zaprezentowanego w filmie:

Plik o rozmiarze 255.00 B w języku polskim

Kod programu (program sprawdzający, czy wskazany rok jest rokiem przestępnym) zaprezentowanego w filmie:

Plik o rozmiarze 150.00 B w języku polskim

Przeczytaj

Wyrażenia (ang. *expressions*) to konstrukcje pozwalające przetwarzać informacje w języku Python. W programowaniu mówimy przede wszystkim o wyrażeniach matematycznych, które służą obliczeniom (jak dodawanie czy mnożenie) oraz logicznych, które zwracają wartość `Prawda/Fałsz`. Musimy również pamiętać o **priorytecie** operatorów.

Priorytet operatorów określa, jak należy interpretować wyrażenia w języku Python. Wyrażenie przetwarzane jest od lewej do prawej strony (oprócz potęgowania, które grupowane jest od prawej do lewej), z zachowaniem reguł matematyki, a więc w kolejności: nawiasy, potęgowanie, pierwiastkowanie, mnożenie, dzielenie itd.

Poniżej znajdziesz wykaz wybranych operatorów wykorzystywanych do budowania wyrażeń w języku Python 3 (im niżej dany operator znajduje się na liście, tym wyższy ma priorytet; wyjątkiem są operatory relacyjne, które mają taki sam priorytet).

Przykłady operatorów arytmetycznych:

Przykład	**Opis**
`x + y`	dodawanie
`x - y`	odejmowanie
`x * y`	mnożenie
`x / y`	dzielenie rzeczywiste
`x // y`	dzielenie całkowite
`x % y`	reszta z dzielenia całkowitego (operacja modulo)
`x ** y`	potęgowanie

Przykłady operatorów relacji:

Przykład	**Opis**
`x < y, x > y`	operatory porównania mniejszy/większy
`x <= y, x >= y`	operatory porównania mniejszy/większy lub równy
`x == y, x != y`	operatory równości i nierówności zawartości
`x in y, x not in y`	przynależność obiektów

Ważne!

W języku Python wszystkie operacje porównań mają taki sam priorytet, który jest niższy od priorytetu wszystkich operacji arytmetycznych.

Przykłady operatorów logicznych:

Przykład	Opis
`x or y`	logiczne OR
`x and y`	logiczne AND
`not x`	logiczna negacja

Dla zainteresowanych

Istnieje jeszcze tzw. `trójargumentowe wyrażenie wyboru`, które pozwala na sprawdzanie pewnych warunków w analogiczny sposób jak w konstrukcji instrukcji warunkowej, lecz zapisywane jest w jednej linii programu. Czasami takie rozwiązanie ma swoje zalety. Sprawdź na konkretnym przykładzie, jak działa w praktyce (zachęcamy do samodzielnych prób):

```
Python 3.8.10 (default, Jun  2 2021, 10:49:15)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license()" for more infc
>>> wiek = 18
>>> dorosly = True if wiek == 18 else "Jeszcze nie"
>>> print(dorosly)
True
>>> wiek = 17
>>> dorosly = True if wiek == 18 else "Jeszcze nie"
>>> print(dorosly)
Jeszcze nie
>>>
```

Dla zainteresowanych

W języku Python istnieją także inne operatory, których będziemy używać w kolejnych e-materiałach.

Przykład	Opis
`lambda argumenty: wyrażenie`	generator funkcji anonimowej
`x[i]`	indeksowanie
`x[i:j]`	wycinki
`x.wlasciwosc`	odwołanie do właściwości obiektu

Oto przykład wykorzystania różnych operatorów w środowisku IDLE dla różnych obiektów:

```
Python 3.6.7 (default, Oct 22 2018, 11:32:17)
[GCC 8.2.0] on linux
Type "help", "copyright", "credits" or "license()" for more inform
>>> portfel = 13500
>>> portfel >= 10000
True
>>> imie = 'Adam'
>>> skarbonka = 1345.26
>>> lista_1 = [1, 34, 35]
>>> lista_2 = [1, 34, 35]
>>> lista_3 = lista_1
>>>
>>> lista_1 == lista_2
True
>>> lista_1 is lista_2
False
>>> lista_1 is lista_3
True
>>> imie > skarbonka
Traceback (most recent call last):
    File "<pyshell#18>", line 1, in <module>
    imie > skarbonka
TypeError: '>' not supported between instances of 'str' and 'float
>>>
```

> **Ważne!**
>
> Język Python operuje **silnymi** typami danych – związane z nimi operacje są dokładnie sprawdzane. I tak np. porównanie nie zawsze jest możliwe: dla obiektów o różnych typach może wystąpić błąd
> `TypeError: '>' not supported between instances of 'str' and 'float'`
> .

Polecenie 1

Wykonaj kilka operacji porównywania wartości stałych lub zmiennych oraz kilka operacji arytmetycznych na różnych typach danych.

Reguła: kolejność działań bez nawiasów

> Wyrażenia bez nawiasów wykonywane są według priorytetów operatorów – np. wyrażenie
> `Q ** 4 + 3 * 5 / 3 - 1`
> możemy zapisać w następujący sposób:
> `((Q ** 4) + (3 * 5/3)) - 1.`
> **Reguła: kolejność działań z nawiasami**
>
> Nawiasy zmieniają kolejność wykonywania działań - np. wyrażenie
> `(Q ** 4 + 3 * 5/(3 - 1))`
> zostanie uproszczone do postaci
> `(Q ** 4 + 7,5)`,
> ponieważ
> `(3 - 1)`
> zapisane jest w nawiasie, a zatem zostanie obliczone przed wykonaniem dzielenia.

Oto przykład w środowisku IDLE:

```
Python 3.6.7 (default, Oct 22 2018, 11:32:17)
[GCC 8.2.0] on linux
Type "help", "copyright", "credits" or "license()" for more inform
>>> Q = 3
>>> Q ** 4 + 3 * 5 / 3 - 1
85.0
>>> # teraz wyrażenie z użyciem nawiasów
>>> Q ** 4 + 3 * 5 / (3 - 1)
88.5
>>>
```

W języku Python typy danych są konwertowane. Opiszemy to na przykładzie dodawania dwóch różnych typów liczbowych, np. `int + float`. Kolejność operacji w języku Python będzie następująca:

- obiekt typu `int` zostanie zamieniony na typ `float`
- wykonane zostanie dodawanie dwóch obiektów tego samego typu `float`
- wynikiem będzie liczba typu `float`.

Oto przykład w środowisku IDLE.

```
Python 3.6.7 (default, Oct 22 2018, 11:32:17)
[GCC 8.2.0] on linux
Type "help", "copyright", "credits" or "license()" for more inform
>>> obiekt_1 = 30
```

```
>>> type(obiekt_1)
<class 'int'>
>>> obiekt_2 = 3.14
>>> type(obiekt_2)
<class 'float'>
>>> obiekt_3 = obiekt_1 + obiekt_2
>>> obiekt_3
33.14
>>> type(obiekt_3)
<class 'float'>
>>>
```

W języku Python istnieje możliwość wygenerowania elementów listy poprzez zapisanie specyficznego kodu – jest to tzw. **wyrażenie listowe**. Jego ogólna postać wygląda następująco (`if warunek` jest opcjonalny):

```
[ wyrażenie for wyraz in sekwencja [if warunek] ]
```

Oto przykład wykonania takiego wyrażania listowego w środowisku IDLE – tworzymy listę kwadratów liczb naturalnych z przedziału `<0,9>`:

```
Python 3.6.7 (default, Oct 22 2018, 11:32:17)
[GCC 8.2.0] on linux
Type "help", "copyright", "credits" or "license()" for more inform
>>> kwadraty = [x ** 2 for x in range(10)]
>>> kwadraty
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
>>>
```

Polecenie 2

Dla x należącego do przedziału $\langle 0, 14 \rangle$ wygeneruj opisane poniżej listy. Przyjmij, że x jest liczbą całkowitą.

1. $(x + 3)^2 \cdot 3$,
2. $(2 + x) \cdot 3$,
3. $2 + x \cdot 3$.

Moduły w języku Python

Polecenie import umożliwia rozszerzenie programu o funkcje zawarte w zewnętrznych modułach. Mogą to być gotowe moduły dostępne w bazie Python Package Index lub napisane przez programistę fragmenty kodu. Przykładowo moduł `math` zawiera takie funkcje matematyczne, jak sinus, pierwiastek kwadratowy, a także wartości stałe, np. stałą `e` lub `pi`.

Polecenie 3

Użyj polecenia `import math` – spowoduje to zaimportowanie modułu zawierającego funkcje podobne do tych, którymi dysponuje kalkulator oraz niektóre stałe matematyczne. Następnie sprawdź zawartość tego modułu za pomocą polecenia `dir(math)` i wykonaj `help(math.sqrt)` oraz `help(math.pi)`.

Przykłady modułów standardowych, dostępnych po instalacji języka Python:

Nazwa	Opis
`math`	funkcje matematyczne
`turtle`	operacje grafiki żółwia LOGO
`json`	moduł wymiany danych
`pickle`	moduł zapisywania wartości zmiennych w plikach

Przykłady modułów dostępnych z Python Package Index:

Nazwa	Opis
`pandas`	obliczenia statystyczne

Nazwa	Opis
`pysimplegui`	moduł do szybkiego tworzenia aplikacji GUI
`pytechbrain`	obsługa mikrokontrolerów Arduino

Ciekawostka

W języku Python dostępna jest ogromna baza gotowych modułów — ponad 100 tys. projektów. Wszystkie możemy przeglądać na stronie Python Package Index.

Słownik

import modułu

moduł to plik zawierający definicje funkcji i stałych w języku Python; wykonanie polecenia `import modul` powoduje dostęp do nich z poziomu głównego skryptu

inspektor obiektów

w środowisku PyCharm element umożliwiający podgląd w czasie rzeczywistym obiektów, ich wartości oraz typu

konwersja

zmiana typu wartości obiektu (np. z `str` na `int` czy z `float` na `str`), w języku Python wykonywana poprzez skopiowanie wartości do innego obiektu

Python Package Index

(w skrócie zapisywane jako PyPI) repozytorium dodatkowych modułów dla języka Python; instalujemy je, korzystając z mechanizmu `pip`

Sprawdź się

Pokaż ćwiczenia:

Ćwiczenie 1

Połącz każdy operator z właściwym opisem.

`x * y`	indeksowanie
`x - y`	logiczna koniunkcja
`x if y else z`	wycinki
`x + y`	operatory porównania
`X ** y`	dzielenie
`not x`	odejmowanie
`x / y`	trójargumentowe wyrażenie wyboru
`x in y, x not in y`	odwołanie do właściwości obiektu
`x < y, x <= y, x > y, x >= y`	logiczna alternatywa
`x == y, x != y`	potęgowanie
`x.wlasciwosc`	mnożenie
`x or y`	przynależność obiektów
`x[i]`	dodawanie
x and y	logiczna negacja
`x[i:j]`	operatory równości i nierówności zawartości

Ćwiczenie 2

Uporządkuj wyrażenia w języku Python, rozpoczynając od tego z największym priorytetem operatora.

`x + y`

`x == y, x != y`

`x * y`

`x[i]`

`x if y else z`

Ćwiczenie 3

Wskaż listę, którą wygeneruje następujące wyrażenie listowe: `[x**3 for x in range(-10, 6, 3)]`.

- ◯ `[-729, -343, -125, -27, -1, 1, 27]`
- ◯ `[-1000, -343, -125, -27, 27, 125]`
- ◯ `[-1000, -343, -64, -1, 8, 125]`
- ◯ `[100, 49, 16, 1, 4, 25]`

Ćwiczenie 4

Uzupełnij zdanie właściwymi elementami.

Operator służący do przeszukiwania elementu w złożonym obiekcie to [ ], a operator przeciwny to [ ].

`not in` `$` `#` `!=` `inside` `in` `@`

Ćwiczenie 5

Dopasuj opisy modułów do ich nazw.

math	moduł obliczeń statystycznych
pandas	proste aplikacje GUI
pysimplegui	moduł grafiki żółwia (LOGO)
JSON	moduł wymiany danych
pytechbrain	moduł funkcji matematycznych
turtle	obsługa mikrokontrolerów Arduino

Ćwiczenie 6

Zaznacz wszystkie poprawne odpowiedzi. Wskaż właściwe sposoby importowania modułów.

- [ ] import moduł as nazwa
- [ ] import moduł
- [ ] from moduł import *
- [ ] import from moduł *
- [ ] from moduł import moduł

Ćwiczenie 7

Wskaż poprawny zapis.

```
# naszym zadaniem jest obliczenie pierwiastka kwadratowego

num1 = 1.5
num2 = ?
```

- ○
```
import math
num2 = sqrt(num1)
```

- ○
```
from math import sqrt
num2 = sqrt(num1)
```

- ○
```
from math import sqrt()
num2 = sqrt(num1)
```

Ćwiczenie 8

Przypisz wartości logiczne `True` lub `False` w zależności od wyników porównań.

`[1, 2, 3] == [1, 2, 3]`	`[1, 2, 3] == [1, 3, 2]`	`[1, 2, 3] < [1, 3, 2]`

`True` `False` `False` `True` `False` `True`

Dla nauczyciela

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

Przedmiot: Informatyka

Temat: Operatory w języku Python

Grupa docelowa: III etap edukacyjny, liceum, 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:
>
> 1) projektuje i programuje rozwiązania problemów z różnych dziedzin, stosuje przy tym: instrukcje wejścia/wyjścia, wyrażenia arytmetyczne i logiczne, instrukcje warunkowe, instrukcje iteracyjne, funkcje z parametrami i bez parametrów, testuje poprawność programów dla różnych danych; w szczególności programuje algorytmy z punktu I.2);
>
> 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 w zakresie rozumienia i tworzenia informacji,
- kompetencje w zakresie wielojęzyczności,
- kompetencje cyfrowe,

- kompetencje osobiste, społeczne i w zakresie umiejętności uczenia się.

Cele operacyjne:

- Zbudujesz wyrażenia w języku programowania.
- Scharakteryzujesz operatory arytmetyczne i logiczne oraz ich priorytety w języku Python.
- Przećwiczysz obliczanie wartości wyrażeń.
- Przygotujesz program z wykorzystaniem modułów języka Python.

Strategie nauczania:

- bezpośrednia strategia poznawcza;
- strukturalna.

Metody i techniki nauczania:

- pogadanka;
- dyskusja;
- metoda przypadków.

Formy pracy:

- praca indywidualna;
- praca w grupach.

Środki dydaktyczne:

- komputery ze środowiskiem Python 3 / IDLE / PyCharm;
- zasoby multimedialne zawarte w e-materiale;
- tablica interaktywna/tablica, pisak/kreda.

Przebieg zajęć:

Faza wstępna

1. Krótka pogadanka na temat kursów walut.
2. Nauczyciel przedstawia temat i cele lekcji.
3. Nauczyciel przedstawia uczniom zadanie do wykonania: Ile można zakupić USD, CHF i EUR za posiadane PLN, przy założeniu, że 40% kwoty PLN ma być przeznaczone na EUR, 25% na CHF a 35% na USD. Uczniowie w parach przygotowują wyjaśnienie problemu.

Faza realizacyjna

1. Praca w parach. Uczniowie tworzą i testują własne wyrażenia algebraiczne obliczające wyniki na podstawie przykładów z e-materiału. Prezentacja wyników na forum klasy.

2. Wspólna dyskusja o zastosowanych operatorach i wyrażeniach.
3. Uczniowie uruchamiają program wykorzystujący zbudowane przez nich wyrażenia.

Faza podsumowująca

1. Wybrany uczeń podsumowuje lekcję, wskazując na zdobyte umiejętności.
2. Nauczyciel omawia przebieg zajęć, wskazuje mocne i słabe strony pracy uczniów, udzielając im tym samym informacji zwrotnej.

Praca domowa:

- Napisz program, który wygeneruje listę 10 wartości w CHF przeliczonych wg aktualnego kursu średniego NBP dla PLN: od 12 zł do 48 zł co 4 zł, a więc dla: 12, 16, 20, 24, 28, 32, 36, 40, 44, 48;
- napisz funkcję, której wynikiem jest lista wygenerowana w poprzednim zadaniu;
- znajdź w serwisie PyPi moduł, który jest zgodny z twoimi zainteresowaniami, np.`Simple client for the Free Internet Chess Server` - zaprezentujesz go klasie.

Materiały pomocnicze:

- Dokumentacja dla Python 3.
- Tabela kursów walut A w serwisie NBP.

Wskazówki metodyczne opisujące różne zastosowania multimedium:

E-materiały dotyczące operatorów mogą stanowić powtórzenie wiadomości ze szkoły podstawowej.