

Jak zmienia się wartość zmiennej w Pythonie

Poradnik z przykładami i odpowiedziami

Nauczysz się odczytywać aktualną wartość zmiennej po kolejnych instrukcjach. Najważniejsza zasada: Python najpierw oblicza prawą stronę przypisania, a potem przypisuje wynik nazwie po lewej stronie.

Co robi operator przypisania

```
x = 5
```

Znak = to operator przypisania. Powyższa instrukcja przypisuje zmiennej x wartość 5. W Pythonie nie trzeba wcześniej osobno deklarować zmiennej ani podawać jej typu.

Zapis czytamy: „Przypisz zmiennej x wartość 5”. Nie jest to pytanie o równość. Do porównywania wartości służy ==, którego w tych przykładach nie potrzebujemy.

Nowa wartość zastępuje poprzednią

```
x = 5  
print(x)  
x = 12  
print(x)
```

Program wypisze kolejno:

```
5  
12
```

Po wykonaniu x = 12 zmienna x ma wartość 12. Poprzednia wartość 5 nie jest już dostępna pod nazwą x. Program wykonuje te instrukcje od góry do dołu.

Przypisanie tekstu

```
slovo = "kot"  
slovo = "pies"  
print(slovo)
```

Wynik to pies. Tekst zapisujemy w cudzysłowie. Sama instrukcja przypisania nie wypisuje wyniku — robi to print().

Zmiana wartości na podstawie innych wartości

Jak działa `x = x + 1`

```
x = 7
x = x + 1
print(x)
```

Najpierw Python odczytuje dotychczasową wartość `x`, czyli 7. Następnie oblicza `7 + 1`, otrzymując 8. Dopiero wtedy przypisuje wynik do `x`. Program wypisze 8.

Po lewej stronie `x` oznacza nazwę, której przypisujemy wynik. Po prawej stronie `x` oznacza jej wartość sprzed wykonania tej instrukcji. Zmienna musi wcześniej otrzymać wartość.

Jak działa `x = b`

```
x = 7
x = x + 1
b = 20
x = b
b = 3
print(x, b)
```

Po instrukcji	x	b
<code>x = 7</code>	7	—
<code>x = x + 1</code>	8	—
<code>b = 20</code>	8	20
<code>x = b</code>	20	20
<code>b = 3</code>	20	3

Wynik: 20 3. Instrukcja `x = b` przypisuje `x` aktualną wartość `b`, czyli 20. Późniejsze `b = 3` nie zmienia `x`. Dla używanych tu liczb przypisanie nie tworzy stałego połączenia między zmiennymi. Symbol — oznacza, że zmienna nie ma jeszcze przypisanej wartości.

Jak analizować program

Czytaj po jednej instrukcji. Oblicz prawą stronę, korzystając z aktualnych wartości. Zmień tylko zmienną po lewej stronie. Zapisuj nowe wartości w tabeli; nie wracaj do wcześniejszych obliczeń. W `print(a, b)` wynik odczytuj w kolejności `a`, potem `b`.

Przykład 1 Kilka kolejnych przypisań

Przewidź wynik `print()`. Każdy przykład uruchamiamy osobno. Tabela pokazuje wartości po kolejnych przypisaniach.

```
a = 4
b = 7
c = a + b
d = c - a
e = d * 2
a = a + 1
b = e - c
c = a + b + d
d = c - e
e = a + c - d
print(a, b, c, d, e)
```

Instrukcja	a	b	c	d	e
a = 4	4	—	—	—	—
b = 7	4	7	—	—	—
c = a + b	4	7	11	—	—
d = c - a	4	7	11	7	—
e = d * 2	4	7	11	7	14
a = a + 1	5	7	11	7	14
b = e - c	5	3	11	7	14
c = a + b + d	5	3	15	7	14
d = c - e	5	3	15	1	14
e = a + c - d	5	3	15	1	19

Po zmianie `a` na 5 wartość `c` nie przelicza się sama. Przy `b = e - c` używamy `e = 14` i `c = 11`. Dopiero kolejna instrukcja nadaje `c` nową wartość. W ostatnim przypisaniu `e` korzystamy już z nowego `d`.

Przykład 2 Przypisanie nie zamienia wartości

Przewidź wynik `print()`. Każdy przykład uruchamiamy osobno. Tabela pokazuje wartości po kolejnych przypisaniach.

```
a = 3
b = 8
c = 2
d = a + b
e = d * c
f = e - b
a = b
b = a
c = f - d
d = a + c
e = e - d
f = e + b - c
print(a, b, c, d, e, f)
```

Instrukcja	a	b	c	d	e	f
a = 3	3	—	—	—	—	—
b = 8	3	8	—	—	—	—
c = 2	3	8	2	—	—	—
d = a + b	3	8	2	11	—	—
e = d * c	3	8	2	11	22	—
f = e - b	3	8	2	11	22	14
a = b	8	8	2	11	22	14
b = a	8	8	2	11	22	14
c = f - d	8	8	3	11	22	14
d = a + c	8	8	3	11	22	14
e = e - d	8	8	3	11	11	14
f = e + b - c	8	8	3	11	11	16

Po `a = b` zmienna `a` ma już wartość 8. Następne `b = a` przypisuje więc `b` również 8. Te dwie linie nie zamieniają wartości 3 i 8. Przy `e = e - d` prawa strona używa starego `e = 22` i nowego `d = 11`.

Przykład 3 Zmienna pomocnicza i wiele zmian

Przewidź wynik `print()`. Każdy przykład uruchamiamy osobno. Tabela pokazuje wartości po kolejnych przypisaniach.

```
x = 5
b = 9
c = 3
d = x + c
e = b - c
temp = x
x = b
b = temp
c = x - b + c
d = d + c
e = d - e
x = x + 1
b = e - b
x = b
c = x + d - c
print(x, b, c, d, e, temp)
```

Instrukcja	x	b	c	d	e	temp
x = 5	5	—	—	—	—	—
b = 9	5	9	—	—	—	—
c = 3	5	9	3	—	—	—
d = x + c	5	9	3	8	—	—
e = b - c	5	9	3	8	6	—
temp = x	5	9	3	8	6	5
x = b	9	9	3	8	6	5
b = temp	9	5	3	8	6	5
c = x - b + c	9	5	7	8	6	5
d = d + c	9	5	7	15	6	5
e = d - e	9	5	7	15	9	5
x = x + 1	10	5	7	15	9	5
b = e - b	10	4	7	15	9	5
x = b	4	4	7	15	9	5
c = x + d - c	4	4	12	15	9	5

Zmienna `temp` zachowuje początkowe `x = 5`. Dzięki temu `x = b` oraz `b = temp` zamieniają wartości `x` i `b`. Później `x = x + 1` daje 10, ale `x = b` zastępuje tę wartość aktualnym `b = 4`. Ostatnie przypisanie do `c` korzysta z `x = 4`, `d = 15` i dotychczasowego `c = 7`.

Poprawne odpowiedzi

Poniższe wiersze to dokładna zawartość wyświetlana przez `print()`. Przecinki w kodzie rozdzielają argumenty funkcji; w wyniku `print()` domyślnie wstawia między nimi spacje.

Przykład 1

Kolejność zmiennych: a, b, c, d, e.

```
5 3 15 1 19
```

Końcowe wartości: a = 5, b = 3, c = 15, d = 1, e = 19.

Przykład 2

Kolejność zmiennych: a, b, c, d, e, f.

```
8 8 3 11 11 16
```

Końcowe wartości: a = 8, b = 8, c = 3, d = 11, e = 11, f = 16.

Przykład 3

Kolejność zmiennych: x, b, c, d, e, temp.

```
4 4 12 15 9 5
```

Końcowe wartości: x = 4, b = 4, c = 12, d = 15, e = 9, temp = 5.

Co warto zapamiętać

Operator `=` przypisuje wartość. Prawą stronę obliczamy przed zmianą lewej. Nowe przypisanie zastępuje poprzednią wartość danej zmiennej. Przy liczbach `x = b` nie sprawia, że późniejsze zmiany b automatycznie zmieniają x. Liczy się kolejność instrukcji.