MATERIAŁY DLA UCZNIA

Python od zera, zmienne i dane od użytkownika

Programowanie | Klasy VII i VIII | 45 minut | Lekcja otwarta

Uczeń pisze w Pythonie program, który pobiera dane od użytkownika, zapisuje je w zmiennych i wypisuje policzony wynik.

Czego się nauczysz

Zmienne i typy danych
Zapisujesz dane w zmiennych i rozróżniasz tekst od liczby.

Dane wejściowe i wynik
Pobierasz dane funkcją input, zamieniasz je na liczbę i wypisujesz wynik funkcją print.

Program w Pythonie to lista poleceń wykonywanych po kolei, od góry do dołu. Żeby napisać coś użytecznego, wystarczą cztery rzeczy: zmienna, print, input i zamiana tekstu na liczbę. Z tego materiału nauczysz się każdej z nich osobno i dowiesz się, dlaczego program, który pyta o dwie liczby i wypisuje 78 zamiast 15, wcale się nie zepsuł.

Zmienna, czyli pudełko z nazwą

Zmienna to miejsce w pamięci, do którego zapisujesz wartość i któremu nadajesz nazwę, żeby móc do niej wrócić. Zapis wygląda tak: cena = 12. Po lewej stronie stoi nazwa, po prawej wartość, a jeden znak równości oznacza polecenie „zapisz to po prawej do tego po lewej”. To nie jest równanie z matematyki i nie znaczy, że cena równa się 12. Znaczy: włóż liczbę 12 do pudełka o nazwie cena.

Zmienna trzyma jedną wartość naraz. Gdy napiszesz cena = 12, a linijkę niżej cena = 20, stara wartość znika bezpowrotnie i w pudełku zostaje 20. Ta kolejność jest ważna, bo Python wykonuje linie po kolei i zawsze używa tego, co w zmiennej jest w danym momencie.

Dwa znaki równości to co innego. Zapis cena == 12 nie zapisuje niczego, tylko pyta, czy w zmiennej cena siedzi 12, i daje odpowiedź prawda albo fałsz. To częste źródło pomyłek na początku.

Nazwa zmiennej podlega kilku regułom. Nie może zaczynać się od cyfry, nie może zawierać spacji, lepiej nie używać w niej polskich znaków. Poza tym Python rozróżnia wielkie i małe litery, więc Cena i cena to dwie różne zmienne. Wybieraj nazwy, które mówią, co jest w środku: cena_biletu jest lepsze niż x, bo za tydzień nadal będziesz wiedzieć, o co chodziło.

SPRÓBUJ SAM

Wpisz w edytorze trzy linie i uruchom: cena = 12, potem cena = 20, potem print(cena). Zastanów się przed uruchomieniem, co się wypisze, a dopiero potem sprawdź. Następnie zamień kolejność dwóch pierwszych linii i uruchom ponownie.

Tekst czy liczba, czyli typy danych

Python rozróżnia rodzaje wartości i nazywa je typami. Na tej lekcji potrzebujesz trzech. Tekst, czyli napis, zapisujesz w cudzysłowie: "Kasia". Liczbę całkowitą, czyli int, zapisujesz bez cudzysłowu: 24. Liczbę z częścią dziesiętną, czyli float, zapisujesz z kropką, nie z przecinkiem: 12.50.

Typ decyduje o tym, co da się z wartością zrobić. Liczby się dodają, mnożą i dzielą. Teksty też można dodać, ale dodawanie tekstów oznacza sklejanie: "7" + "8" daje "78", bo Python nie liczy, tylko dostawia jeden napis za drugim. To nie jest błąd programu, tylko poprawne zachowanie zgodne z typem.

Ta sama cyfra może być jednym albo drugim, w zależności od zapisu. 5 to liczba, a "5" to tekst zawierający znak pięć. Wyglądają na ekranie tak samo, a zachowują się zupełnie inaczej. Dlatego przy każdej wartości warto sobie odpowiedzieć na jedno pytanie: czy będę tym liczyć. Jeśli tak, potrzebujesz liczby.

Int czy float wybierasz według tego, czy wartość ma część dziesiętną. Liczba uczniów to int, bo pół ucznia nie istnieje. Cena z groszami to float, bo int obciąłby grosze.

SPRÓBUJ SAM

Uruchom kolejno dwie linie: print(7 + 8), a potem print("7" + "8"). Zapisz oba wyniki obok siebie i napisz jednym zdaniem, czym różnią się te dwa działania, choć znak dodawania jest w obu ten sam.

print i input, czyli rozmowa z użytkownikiem

Funkcja print wypisuje na ekranie to, co dostanie w nawiasie. Bez niej program nadal liczy, tylko po cichu, i nie zobaczysz żadnego wyniku. To częste rozczarowanie: program działa poprawnie, a ekran jest pusty, bo nikt nie kazał niczego wypisać.

W nawiasie print możesz podać tekst w cudzysłowie, zmienną bez cudzysłowu albo kilka rzeczy po przecinku. Zapis print("Koszt to", koszt, "zlotych") wypisze wszystkie trzy elementy w jednej linii, oddzielone spacjami. Uważaj na cudzysłowy: print("koszt") wypisze słowo koszt, a print(koszt) wypisze zawartość zmiennej. To dwa różne polecenia.

Funkcja input działa odwrotnie: zatrzymuje program i czeka, aż użytkownik coś wpisze i naciśnie Enter. Tekst podany w nawiasie input pojawia się jako zachęta, na przykład input("Podaj swoje imie: "). To, co użytkownik wpisze, trzeba od razu zapisać do zmiennej, inaczej wartość przepadnie: imie = input("Podaj swoje imie: ").

Jest jedna rzecz o input, którą trzeba zapamiętać na zawsze i którą opisuje następna sekcja. Input zawsze oddaje tekst. Zawsze, nawet gdy użytkownik wpisał same cyfry.

SPRÓBUJ SAM

Napisz program z dwóch linii: imie = input("Podaj swoje imie: ") oraz print("Czesc", imie, "dobrze Cie widziec"). Uruchom go i wpisz swoje imię. Potem usuń cudzysłowy wokół słowa Czesc i uruchom ponownie. Przeczytaj komunikat, który się pojawi.

Zamiana tekstu na liczbę, czyli int i float

Skoro input zawsze oddaje tekst, to program, który pyta o dwie liczby i je dodaje, bez dodatkowego kroku nie policzy sumy. Użytkownik wpisuje 7 i 8, ale do zmiennych trafiają napisy "7" i "8", a dodawanie napisów skleja je w "78". To najczęstsza pułapka pierwszych programów i wcale nie kończy się komunikatem błędu, więc łatwo jej nie zauważyć.

Rozwiązaniem jest jawna zamiana typu. Funkcja int bierze tekst i oddaje liczbę całkowitą: liczba = int(input("Podaj liczbe: ")). Funkcja float robi to samo dla liczb z częścią dziesiętną. Zapis czyta się od środka na zewnątrz: najpierw wykonuje się input, potem jego wynik trafia do int.

Jeśli spróbujesz liczyć na tekście bez zamiany, na przykład pomnożyć napis przez liczbę zmiennoprzecinkową, Python zatrzyma program i pokaże TypeError. Ten komunikat oznacza dokładnie jedno: użyłeś typu, który do tego działania nie pasuje. To nie jest awaria, tylko informacja, gdzie brakuje int albo float.

Działa to też w drugą stronę. Jeśli użytkownik wpisze słowo zamiast cyfry, int nie ma czego zamienić i program zatrzyma się z komunikatem ValueError. Na razie wystarczy wiedzieć, skąd on się bierze.

SPRÓBUJ SAM

Napisz program: a = int(input("Pierwsza liczba: ")), b = int(input("Druga liczba: ")), print(a + b). Uruchom i wpisz 7 oraz 8. Potem usuń obie funkcje int, uruchom ponownie z tymi samymi danymi i zapisz oba wyniki obok siebie razem z wyjaśnieniem różnicy.

Czytanie komunikatu błędu

Komunikat błędu, wyświetlany zwykle na czerwono pod programem, nie jest karą ani znakiem, że nie nadajesz się do programowania. To najbardziej użyteczna informacja, jaką Python może Ci dać, bo zwykle wskazuje palcem miejsce i rodzaj problemu. Warto czytać go w dwóch miejscach: numer linii mówi, gdzie szukać, a ostatnia linijka komunikatu mówi, co jest nie tak.

Trzy rodzaje spotkasz najczęściej. SyntaxError oznacza, że Python nie rozumie zapisu: zwykle brakuje nawiasu zamykającego, cudzysłowu albo dwukropka. Sprawdź wtedy, czy każdy nawias otwierający ma swoją parę. TypeError oznacza działanie na niepasujących typach, czyli najczęściej liczenie na tekście z input. NameError oznacza, że użyłeś nazwy, której Python nie zna: literówka w nazwie zmiennej albo Print zamiast print, bo wielkość liter ma znaczenie.

Warto też wiedzieć, że numer linii bywa przybliżony. Przy brakującym nawiasie Python zauważa problem dopiero w następnej linii, więc sprawdzaj również linię powyżej wskazanej.

Najlepszy nawyk na początek to uruchamianie programu po każdej dopisanej linii, a nie po napisaniu całości. Kiedy dopisujesz jedną linię i program przestaje działać, wiesz dokładnie, która linia zawiniła. Kiedy dopisujesz dwadzieścia, zaczyna się zgadywanie.

SPRÓBUJ SAM

Zepsuj celowo działający program na trzy sposoby, za każdym razem na chwilę: usuń jeden nawias zamykający, napisz Print wielką literą, wykonaj dodawanie na wartości z input bez int. Zapisz nazwę błędu i numer linii dla każdego przypadku. Zbuduj z tego własną ściągę, którą sprawdzisz przy następnym programie.

Słowniczek

zmienna
Nazwane miejsce w pamięci, do którego zapisujesz jedną wartość i możesz ją później odczytać albo nadpisać.

przypisanie
Zapis z jednym znakiem równości, który wkłada wartość z prawej strony do zmiennej z lewej strony.

typ danych
Rodzaj wartości decydujący o tym, co można z nią zrobić: tekst, liczba całkowita albo liczba z częścią dziesiętną.

int
Liczba całkowita, a także funkcja zamieniająca tekst na taką liczbę.

float
Liczba z częścią dziesiętną zapisywaną po kropce, a także funkcja zamieniająca na nią tekst.

print
Funkcja wypisująca na ekranie to, co dostanie w nawiasie.

input
Funkcja, która zatrzymuje program, czeka na dane od użytkownika i zawsze oddaje je jako tekst.

TypeError
Komunikat oznaczający próbę wykonania działania na wartościach niepasującego typu, najczęściej liczenia na tekście.

Częste pomyłki

- Liczenie na wartości prosto z input bez zamiany na liczbę. Dodawanie skleja wtedy napisy i z 7 oraz 8 wychodzi 78 zamiast 15. Owiń input funkcją int albo float, zanim zaczniesz liczyć.
- Użycie dwóch znaków równości przy tworzeniu zmiennej. Jeden znak zapisuje wartość, dwa znaki tylko pytają, czy wartości są równe. Przy tworzeniu zmiennej zawsze jeden.
- Nazwanie zmiennej od cyfry albo ze spacją. Python odmówi wykonania takiej linii i pokaże SyntaxError. Nazwa zaczyna się od litery, nie ma spacji, a słowa łączysz podkreśleniem, na przykład cena_biletu. Polskich znaków unikaj, bo działają, ale łatwo pomylić a z ą i długo szukać literówki.

- Pisanie Print albo INPUT wielkimi literami. Python rozróżnia wielkość liter i nie zna takiej nazwy, więc pokaże NameError. Nazwy funkcji piszemy małymi literami.

Sprawdź, czy rozumiesz

1. Program pyta o dwie liczby i wypisuje 1520 zamiast 35. Czego w nim brakuje i w której dokładnie linii to naprawisz?
2. Kiedy przy pobieraniu ceny użyjesz int, a kiedy float, i co konkretnie stanie się z ceną 12 złotych 50 groszy, jeśli wybierzesz źle?
3. Program wypisuje NameError w linii z print. Wymień dwie różne przyczyny, które mogą dawać ten sam komunikat, i powiedz, jak je od siebie odróżnisz.

Wyzwanie dla chętnych

PONAD PROGRAM

Napisz program, który pyta o liczbę sekund (na przykład 3725) i wypisuje, ile to godzin, minut i sekund. Podpowiedź: operator // dzieli bez reszty, a operator % daje resztę z dzielenia. Sekundy to liczba całkowita, więc użyj int.

Jak sprawdzisz, że Ci wyszło: Dla 3725 program ma wypisać 1 godzinę, 2 minuty i 5 sekund. Dla 59 ma wypisać 0 godzin, 0 minut i 59 sekund. Dla 3600 ma wypisać 1 godzinę, 0 minut i 0 sekund. Jeśli wszystkie trzy się zgadzają, wyszło.

Zadanie do domu na trzy poziomy

Wybierz jeden poziom. Możesz zacząć od łatwego i wrócić po trudniejszy.

POZIOM: ŁATWE

Napisz program, który pyta o Twoje imię i ulubioną liczbę, a potem wypisuje jedno zdanie z obiema wartościami, na przykład: Ola lubi liczbę 7. Liczbę pobierz funkcją int i sprawdź, że program działa także dla liczby 0.

POZIOM: ŚREDNIE

Napisz kalkulator kosztu wyjścia klasowego: cena biletu (float), liczba uczniów (int), koszt przejazdu (float), a na końcu koszt całkowity w jednej czytelnej linii. Sprawdź go dla trzech zestawów danych, w tym dla liczby uczniów równej 0, i zapisz w zeszycie, czy wyniki mają sens.

POZIOM: TRUDNE

Rozbuduj kalkulator o liczbę opiekunów i koszt na jednego ucznia zaokrąglony funkcją round do dwóch miejsc. Następnie celowo wpisz zero uczniów, przeczytaj komunikat ZeroDivisionError i zapisz w komentarzu w kodzie, w której linii i dlaczego program się zatrzymał.

Samoocena

Zaznacz, co już potrafisz. To tylko dla Ciebie, nikt tego nie ocenia.

- [ ] Potrafię utworzyć zmienną z jednym znakiem równości i nadać jej nazwę, która mówi, co jest w środku.
- [ ] Potrafię odróżnić tekst w cudzysłowie od liczby i wybrać int albo float w zależności od tego, czy wartość ma część dziesiętną.
- [ ] Potrafię pobrać dane funkcją input, zamienić je na liczbę funkcją int albo float i wypisać wynik funkcją print.
- [ ] Potrafię przeczytać komunikat błędu, znaleźć numer linii i wyjaśnić, dlaczego program wypisał 78 zamiast 15.