SCENARIUSZ LEKCJI

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 uczy ta lekcja

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.

Potrzebny sprzęt

Laptopy szkolne z przeglądarką, jeden laptop na ucznia lub na parę, projektor albo tablica do pokazu. Internet potrzebny tylko wtedy, gdy pracujecie w edytorze w przeglądarce. Jeśli na laptopach jest zainstalowane Thonny (darmowy edytor Pythona), lekcja działa bez internetu. W przeglądarce użyjcie edytora, który uruchamia Python 3 bez zakładania konta przez ucznia, na przykład Online Python Compiler na stronie programiz.com albo trinket.io w trybie bez logowania. Sprawdźcie dzień wcześniej, że wybrany edytor przyjmuje dane z klawiatury przez input, bo nie każdy edytor online to obsługuje.

Podstawa programowa

Informatyka, Szkoła podstawowa, klasy VII i VIII

Cele kształcenia, wymagania ogólne

- I. Rozumienie, analizowanie i rozwiązywanie problemów na bazie logicznego i abstrakcyjnego myślenia, myślenia algorytmicznego i sposobów reprezentowania informacji.
- 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.1. projektuje, tworzy i testuje programy w procesie rozwiązywania problemów. W programach stosuje: instrukcje wejścia / wyjścia, wyrażenia arytmetyczne i logiczne, instrukcje warunkowe, instrukcje iteracyjne, funkcje oraz zmienne i tablice. W szczególności programuje algorytmy z działu I pkt 2; (Pełne pokrycie celu lekcji. Punkt wymienia wprost instrukcje wejścia / wyjścia oraz zmienne, czyli dokładnie to, co uczeń robi na tej lekcji w Pythonie.)
- I.1. formułuje problem w postaci specyfikacji (czyli opisuje dane i wyniki) oraz wyróżnia kroki w algorytmicznym rozwiązywaniu problemów. Stosuje różne sposoby przedstawiania algorytmów, w tym w języku naturalnym, w postaci listy kroków; (Podstawa dla wątku specyfikacji: uczeń opisuje dane (to, co pobiera od użytkownika) i wyniki (to, co program wypisuje).)

Podstawa prawna: Rozporządzenie Ministra Edukacji Narodowej z dnia 14 lutego 2017 r. w sprawie podstawy programowej wychowania przedszkolnego oraz podstawy programowej kształcenia ogólnego dla szkoły podstawowej, w tym dla uczniów z niepełnosprawnością intelektualną w stopniu umiarkowanym lub znacznym, kształcenia ogólnego dla branżowej szkoły I stopnia, kształcenia ogólnego dla szkoły specjalnej przysposabiającej do pracy oraz kształcenia ogólnego dla szkoły policealnej, Dz.U. 2017 poz. 356.

Brzmienie: Załącznik nr 2 do Dz.U. 2017 poz. 356 w brzmieniu nadanym przez Dz.U. 2024 poz. 996, część KLASY VII i VIII

Lista kontrolna przed lekcją

Do odhaczenia dzień wcześniej i 10 minut przed dzwonkiem.

- [ ] Dzień wcześniej: uruchomcie na jednym szkolnym laptopie program z jedną linią print("test") w edytorze, którego użyje klasa (Thonny albo edytor w przeglądarce bez konta).
- [ ] Dzień wcześniej: sprawdźcie w tym samym edytorze program z input, bo część edytorów online nie przyjmuje danych z klawiatury. Wpiszcie liczbę i upewnijcie się, że program ją odczytał.
- [ ] Dzień wcześniej: wydrukujcie kartę pracy, po jednej na ucznia, a dla wariantu łatwiejszego przygotujcie wydruk programu z trzema lukami (kod jest w kluczu karty pracy, zadanie 3).
- [ ] Dzień wcześniej: zapiszcie w jednym pliku działające wzorce zadań 1 do 3 z klucza karty pracy, żeby przy obchodzie porównywać kod ucznia ze wzorcem.
- [ ] 10 minut przed lekcją: włączcie projektor i otwórzcie edytor w powiększeniu czcionki, tak żeby kod był czytelny z ostatniej ławki (w Thonny: menu Widok, Powiększ).
- [ ] 10 minut przed lekcją: napiszcie na tablicy adres edytora w przeglądarce albo nazwę programu do otwarcia oraz nazwę pliku do zapisania na koniec, na przykład lekcja1_imie.py.
- [ ] 10 minut przed lekcją: przygotujcie na kartce trzy linie z rozgrzewki, żeby nie pisać ich na żywo i nie tracić czasu na start.

- [ ] Na wypadek awarii: przygotujcie po trzy karteczki na ucznia z napisami imie, cena, liczba_osob do wariantu bez sprzętu.

Rozgrzewka na start

BEZ SPRZĘTU, 3 MIN

Pokażcie na projektorze albo napiszcie na tablicy trzy linie i zapytajcie: co pojawi się na ekranie po uruchomieniu?
imie = "Ola"
wiek = 13
print(imie, "ma", wiek, "lat")
Uczniowie odpowiadają z miejsca, bez komputerów. Potem zapytajcie, co się zmieni, jeśli w drugiej linii wpiszecie 14.

Czego się spodziewać: Klasa powinna dojść do wypisu: Ola ma 13 lat. Po zmianie drugiej linii: Ola ma 14 lat. Jeśli ktoś odpowie, że wypisze się słowo imie albo wiek, to dobry moment na zdanie: nazwa bez cudzysłowu oznacza zawartość pudełka, a z cudzysłowem sam napis.

Przebieg lekcji

Pięć etapów, razem 45 minut.

1

Wejście, 5 min

Nauczyciel pokazuje na projektorze trzy linijki kodu w Pythonie (kod jest w polu rozgrzewka) i pyta klasę, co się wypisze. Uczniowie odpowiadają z miejsca, bez komputerów. Nauczyciel podaje cel lekcji: każdy wychodzi z działającym programem, który pyta o dane i liczy wynik. Uczniowie logują się do laptopów i otwierają edytor wskazany przez nauczyciela: Thonny albo edytor w przeglądarce, którego adres jest zapisany na tablicy.

2

Pokaz, 10 min

Nauczyciel pisze na żywo program liczący koszt wycieczki. Krok po kroku: przypisanie do zmiennej, wypisanie zmiennej przez print, pobranie danych przez input, zamiana tekstu na liczbę przez int albo float, wypisanie wyniku. Po każdym kroku uruchamia program, żeby klasa widziała efekt. Nauczyciel celowo popełnia jeden błąd: dodaje wartość prosto z input do liczby, na przykład liczba_uczniow + 1, pokazuje komunikat TypeError i naprawia go, owijając input funkcją int. Zapisuje na tablicy cztery słowa do zapamiętania: zmienna, print, input, int.

3

Praca uczniów, 20 min
Uczniowie pracują z kartą pracy. Zadanie 1: program wypisuje powitanie z imieniem pobranym od użytkownika. Zadanie 2: program pyta o dwie liczby i wypisuje sumę. Zadanie 3: kalkulator kosztu wyjścia klasowego, cena biletu razy liczba uczniów plus koszt przejazdu. Zadania 4 i 5 na karcie pracy to tabela typów i krótkie doświadczenie z usunięciem int, do zrobienia po zadaniu 3. Dla chętnych, którzy skończą wszystko: program liczy, ile minut ma podana liczba godzin i minut. Nauczyciel chodzi po sali, nie poprawia kodu za ucznia, tylko pyta, którą linię uruchomił jako ostatnią i co pokazał komunikat błędu.

4

Sprawdzenie, 6 min
Dwoje uczniów pokazuje swój program na projektorze i wpisuje dane podane przez klasę. Nauczyciel prosi o wskazanie w kodzie miejsca, w którym powstaje zmienna, i miejsca, w którym tekst staje się liczbą. Klasa sprawdza, czy program działa też dla liczby uczniów równej 0. Program z zadania 3 ma wtedy wypisać sam koszt przejazdu.

5

Podsumowanie, 4 min
Nauczyciel wraca do czterech słów z tablicy i prosi, żeby każdy uczeń zapisał w zeszycie jedno zdanie: co robi input i dlaczego potrzebna jest funkcja int. Zapowiada następną lekcję, na której te same zmienne posłużą do zbudowania gry tekstowej. Uczniowie zapisują swój plik pod nazwą wskazaną przez nauczyciela.

Do powiedzenia na głos

„Zmienna to pudełko z nazwą. Wkładacie do niego wartość, a potem wołacie po nazwie i dostajecie to, co w środku.”

„Funkcja input zawsze oddaje tekst, nawet gdy wpisaliście liczbę. Dlatego zanim policzycie, zamieniacie tekst na liczbę funkcją int albo float.”

„Komunikat błędu, ten wyświetlany zwykle na czerwono pod programem, nie znaczy, że zepsuliście komputer. To najkrótsza informacja, w której linii jest problem, więc przeczytajcie ostatnią linijkę komunikatu.”

„Na koniec lekcji każdy ma działający program, który pyta o dane i podaje wynik. Nie o piękny kod chodzi, tylko o to, żeby ruszył.”

Typowe błędy uczniów i co z nimi zrobić

- Uczeń liczy na tekście z funkcji input i dostaje TypeError. Poproś o pokazanie linii z input i wspólnie dopiszcie int wokół niej.
- Uczeń myli przypisanie z porównaniem i pisze dwa znaki równości przy tworzeniu zmiennej. Przypomnij, że jeden znak równości zapisuje wartość do zmiennej.
- Uczeń zapomina nawiasów albo cudzysłowów i dostaje SyntaxError. Pokaż, że Python wskazuje numer linii, i naucz sprawdzać parę otwierającą i zamykającą.
- Uczeń nazywa zmienną od cyfry albo wstawia spację w nazwie i dostaje SyntaxError. Ustalcie zasadę: nazwa zaczyna się od litery, nie ma spacji, słowa łączy podkreślenie. Polskie znaki Python 3 przyjmie, ale odradźcie je, bo łatwo pomylić a z ą i szukać literówki przez pół lekcji.
- Uczeń pisze Print zamiast print. Przypomnij, że Python rozróżnia wielkie i małe litery.

Warianty lekcji

Bez sprzętu i internetu

Pracujecie na kartkach i na tablicy. Każdy uczeń dostaje trzy karteczki podpisane jako zmienne: imie, cena, liczba_osob. Nauczyciel dyktuje instrukcje, a uczniowie wpisują wartości na karteczki i wykreślają stare przy ponownym przypisaniu, żeby zobaczyć, że zmienna trzyma jedną wartość naraz. Potem w parach zapisują w zeszycie program liczący koszt wyjścia klasowego i wykonują go ręcznie dla trzech zestawów danych podanych przez nauczyciela. Na koniec każda para czyta kod na głos, a klasa podaje wynik.

Dla szybszych uczniów

Rozbuduj kalkulator tak, żeby pytał o cenę biletu, liczbę uczniów, liczbę opiekunów i koszt przejazdu, a potem wypisał koszt całkowity oraz koszt na jednego ucznia zaokrąglony do dwóch miejsc funkcją round. Dodaj wypisanie krótkiego podsumowania w jednej linii, na przykład: Wyjście dla 24 uczniów kosztuje 720 złotych, czyli 30 złotych na osobę. Uwaga: przy zerowej liczbie uczniów dzielenie zatrzyma program komunikatem ZeroDivisionError. Zapisz w komentarzu, dlaczego tak się dzieje. Zabezpieczenie warunkiem if poznasz na kolejnych lekcjach.

Przy trudnościach

Dostajesz gotowy program z trzema lukami oznaczonymi komentarzem. Twoje zadanie to wpisać tylko nazwę zmiennej, funkcję int i nazwę zmiennej w print. Zamiast pobierania danych funkcją input możesz na starcie wpisać wartości wprost do zmiennych i uruchomić program, a input dodać dopiero na końcu.

Kryteria oceny

Trzy poziomy dla każdej umiejętności. Opis mówi, co uczeń pokazuje, a nie ile punktów dostaje.

UMIEJĘTNOŚĆ	PODSTAWOWY	DOBRY	BARDZO DOBRY
Zmienne i typy danych	Uczeń tworzy zmienną z jednym znakiem równości, nadaje jej poprawną nazwę (bez spacji, nie od cyfry) i wypisuje jej wartość funkcją print. Rozróżnia tekst w cudzysłowie od liczby bez cudzysłowu.	Uczeń dobiera typ do wartości: int dla liczby uczniów, float dla ceny z groszami. Wyjaśnia, że nowe przypisanie kasuje starą wartość, i przewiduje wynik programu, w którym zmienna zmienia wartość dwa razy.	Uczeń przewiduje bez uruchamiania wynik działań na różnych typach, na przykład sklejenie dwóch tekstów albo powtórzenie tekstu pomnożonego przez liczbę całkowitą, i uzasadnia, dlaczego to nie jest błąd programu. Nazywa zmienne tak, że kod czyta się bez komentarzy.
Dane wejściowe i wynik	Uczeń pobiera dane funkcją input z tekstem zachęty, zapisuje je do zmiennej i wypisuje wynik funkcją print. Powtarza zasadę: input zawsze oddaje tekst.	Uczeń samodzielnie owija input funkcją int albo float i pisze działający program liczący sumę oraz kalkulator kosztu wyjścia klasowego, w którym wynik jest liczony w osobnej zmiennej.	Uczeń czyta komunikat błędu, wskazuje linię i naprawia TypeError bez pomocy nauczyciela. Sprawdza program dla danych brzegowych, na przykład zera, i wyjaśnia, dlaczego wynik 78 zamiast 15 nie kończy się komunikatem błędu.

Pytania do dyskusji

Na podsumowanie lekcji. Nie mają jednej poprawnej odpowiedzi, chodzi o uzasadnienie.

1. Program wypisał 78 zamiast 15 i nie pokazał żadnego błędu. Który błąd jest groźniejszy: taki, który zatrzymuje program, czy taki, który po cichu daje zły wynik? Gdzie w prawdziwym życiu taki cichy błąd mógłby kosztować pieniądze?

2. Komputer nie zgaduje, czy 5 to liczba, czy znak. Czy to wada, czy zaleta? Jak wyglądałoby programowanie, gdyby komputer próbował domyślać się, o co nam chodzi?
3. Nasz kalkulator liczy koszt wyjścia klasowego. Jakie dane wpisane przez użytkownika mogłyby go zmylić (ujemna liczba uczniów, litera zamiast ceny, przecinek zamiast kropki) i kto powinien to sprawdzać: program czy człowiek?

Częste pytania uczniów

Po co zamieniać tekst na liczbę, skoro wpisałem cyfrę?

Bo komputer nie zgaduje, co miałeś na myśli. Input oddaje ciąg znaków, a znak 5 to nie to samo co liczba 5. Funkcja int mówi wprost: potraktuj to jako liczbę.

Czym różni się int od float?

Int to liczba całkowita, float to liczba z częścią dziesiętną. Liczbę uczniów zapisz jako int, cenę z groszami jako float.

Czy mogę nazwać zmienną dowolnie?

Prawie. Nazwa nie może mieć spacji ani zaczynać się od cyfry, a wielkość liter ma znaczenie. Wybieraj nazwy mówiące, co jest w środku, na przykład cena_biletu, a nie x.

Mini projekt na kolejne lekcje

Na 2 do 3 kolejne lekcje albo na kółko. Buduje na tym, co uczniowie zrobili na tej lekcji.

Kalkulator wyjścia klasowego z podsumowaniem

Na dwóch do trzech kolejnych lekcjach albo na kółku uczniowie rozbudowują kalkulator z tej lekcji w narzędzie, którym samorząd klasowy policzy prawdziwe wyjście: do kina, na basen albo na wycieczkę. Projekt buduje na zmiennych, input, int i float, a w kolejnych krokach dokłada zaokrąglanie funkcją round i pierwsze warunki if.

1. Krok 1: kalkulator pyta o cenę biletu, liczbę uczniów, liczbę opiekunów i koszt przejazdu, a potem wypisuje koszt całkowity i koszt na jednego ucznia zaokrąglony funkcją round do dwóch miejsc.

2. Krok 2: uczniowie dodają zmienną z kwotą zebraną do tej pory i program wypisuje, ile jeszcze brakuje albo ile zostanie. Sprawdzają program dla trzech prawdziwych zestawów danych z życia klasy.

3. Krok 3: po poznaniu instrukcji if program sprawdza, czy liczba uczniów jest większa od zera, i zamiast błędu ZeroDivisionError wypisuje czytelny komunikat. Opiekunowie wchodzą bezpłatnie, jeśli klasa tak ustali.

4. Krok 4: pary uczniów wymieniają się programami i testują cudzy kod, wpisując dane, które mogą go zepsuć. Każda para zapisuje, co znalazła, i oddaje autorowi do naprawy.

Co powstaje na końcu: Działający program w jednym pliku, który liczy koszt prawdziwego wyjścia klasowego i podaje kwotę na osobę, oraz krótka lista danych wejściowych, które program obsłużył poprawnie, przetestowana przez inną parę.