Wstęp do programowania

Student po zaliczeniu przedmiotu ma umiejętności, które pozwalają mu:

- świadomie i skutecznie korzystać z zasobów wielokrotnego użycia w konstruowaniu prostych programów komputerowych,
- potrafi zaprojektować, zaimplementować, weryfikować poprawność i debugować proste programy w strukturalnym języku programowania
- implementować podstawowe algorytmy, a także ocenić ich złożoność

Ocenę z przedmiotu Student uzyskuje się na podstawie kolokwium z laboratorium w formie zadań praktycznych.

Treści kształcenia realizowane w ramach laboratorium

Lp.	Tematyka
L1	Uruchamianie programów w języku Python
L2	Interakcja z programem, wyświetlanie danych, wprowadzanie danych do programu
L3	Proste zadania algorytmiczne z wykorzystaniem instrukcji warunkowych i iteracyjnych
L4	Praca z listami i krotkami
L5	Praca z setami i słownikami
L6	Definiowanie funkcji
L7	Wykorzystanie biblioteki Numpy, praca z tablicami
L8	Analiza danych z wykorzystaniem biblioteki Pandas

Laboratorium nr 1

Tematyka : Uruchamianie programów w języku Python, interakcja z programem, wyświetlanie danych, wprowadzanie danych do programu.

Komputery w salach laboratoryjnych mają już zainstalowane: Python 3.12, PyCharm 2024.2 oraz Git 2.35 i ich dodatkowe konfigurowanie nie jest wymagane w trakcie zajęć.

Do pobrania:

Python

https://www.python.org/downloads/

Na swój komputer osobisty pobierz i zainstaluj środowisko Python. Pamiętaj proszę o zaznaczeniu opcji **Add Python to path** w trakcie instalacji, co pozwoli użytkownikowi systemowemu na szybki dostęp do środowiska Python z poziomu konsoli systemowej.

PyCharm

https://www.jetbrains.com/pycharm-edu/

Następnie pobierz PyCharm Community Edition pozwalający na tworzenie projektów w Pythonie.

Git

https://git-scm.com/download/

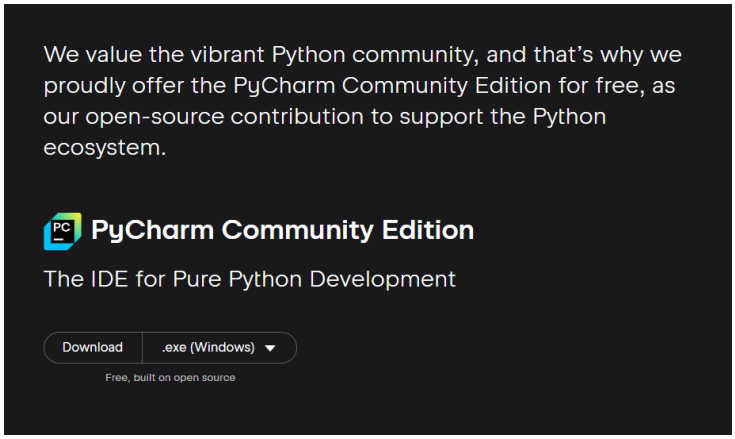
Git jest potrzebny w przypadku wykorzystywania systemu wersjonowania kodu (repozytorium). Dla laboratoriów ze Wstępu do Programowania jest to krok opcjonalny.

Aby móc korzystać z wersjonowania należy najpierw założyć konto na jednym z portali je oferujących, a następnie założyć repozytorium.

Na zajęciach będziemy kożystać repozytoria zamieszczone na GitHub: https://github.com/

Przygotowanie środowiska programistycznego:

Ze strony internetowej https://www.jetbrains.com/pycharm/ należy pobrać i zainstalować IDE (Integrated Development Environment), PyCharm, wybieramy wersję środowiska Community, która jest darmowa, open-sourcowa. **Narzędzie PyCharm służy nam wyłącznie do pisania kodu.**



W kolejnym etapie przechodzimy do oficjalnej strony internetowej języka Pythona https://www.python.org/downloads/ i pobieramy właściwy interpreter, który pozwala nam na pisanie i odtwarzanie plików programów pisanych w środowisko Python. Narzędzie Python.org odpowiada za całą interpretacje kodu. W czasie instalacji należy zaznaczyć opcję Add Python to path, dzięki czemu możliwy jest dostęp do środowiska Python z poziomu konsoli systemowej.

Tworzenie kodu w Pythonie można realizować za pomocą różnych narzędzi, które ułatwiają programowanie, debugowanie, zarządzanie projektami i pracę zespołową np.:.

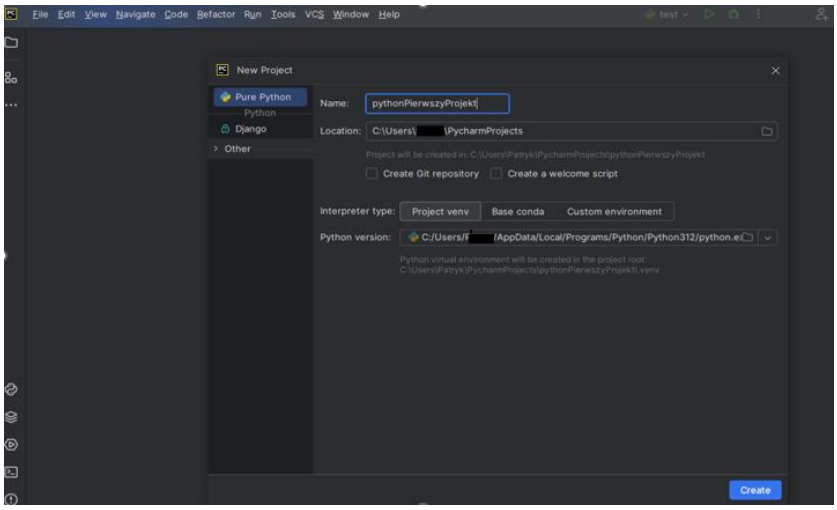
- **Visual Studio Code**: Lekki edytor tekstowy z dużą ilością rozszerzeń, które dodają funkcje takie jak autouzupełnianie, linting, debugowanie i inne.
- **Sublime Text**: Szybki i konfigurowalny edytor tekstowy z wsparciem dla wielu języków programowania.
- **Atom**: Edytor tekstowy stworzony przez GitHub, oferujący wiele wtyczek i rozszerzeń.
- **Jupyter Notebook**: Świetne narzędzie do pracy z kodem Pythonowym, szczególnie w kontekście analizy danych, uczenia maszynowego i nauki

Tworzenie nowego projektu w narzędziu PyCharm

1. Ekran startowy: Jeśli jest to pierwsze uruchomienie PyCharm, zobaczymy ekran powitalny. W przeciwnym razie, wybieramy File > New Project... z paska menu.
2. Konfiguracja nowego projektu:

- Nazwa projektu: Podajemy nazwę swojego projektu (pole: Name).
- Lokalizacja projektu: Wybieramy miejsce na dysku, gdzie projekt ma być zapisany (pole: Location).
- Interpreter: zostanie uzupełniony automatycznie (powinien zostać wskazany plik python.exe). PyCharm powinien automatycznie wykryć interpretery zainstalowane na twoim systemie (pole: Python version).

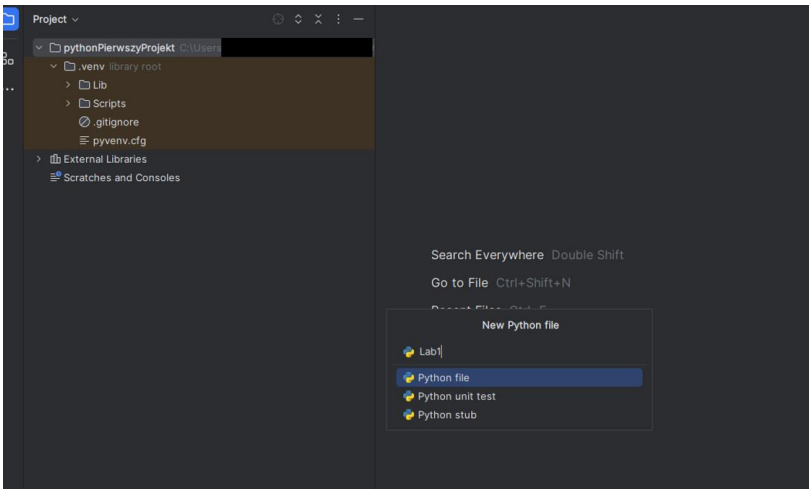
3. Zakończenie tworzenia projektu: Kliknij Create lub Finish, aby zakończyć proces tworzenia projektu.



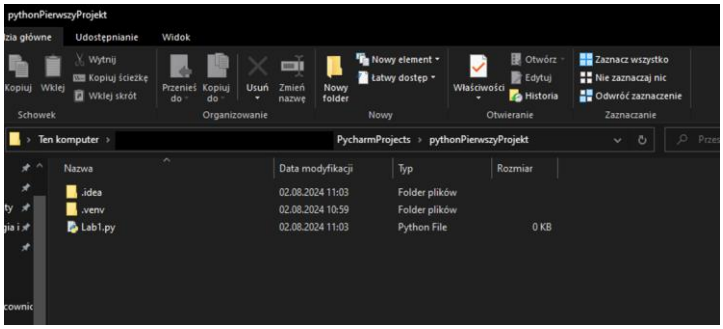
Dodawanie plików i folderów

Po utworzeniu projektu, możemy dodawać nowe pliki i foldery:

1. Klikamy prawym przyciskiem myszy na katalogu projektu w panelu po lewej stronie.
2. Wybieramy New > Python File lub New > Directory w zależności od tego, co chcemy dodać.
3. Praca nad projektem, od teraz możemy zacząć kodować, korzystając z wielu funkcji PyCharm.

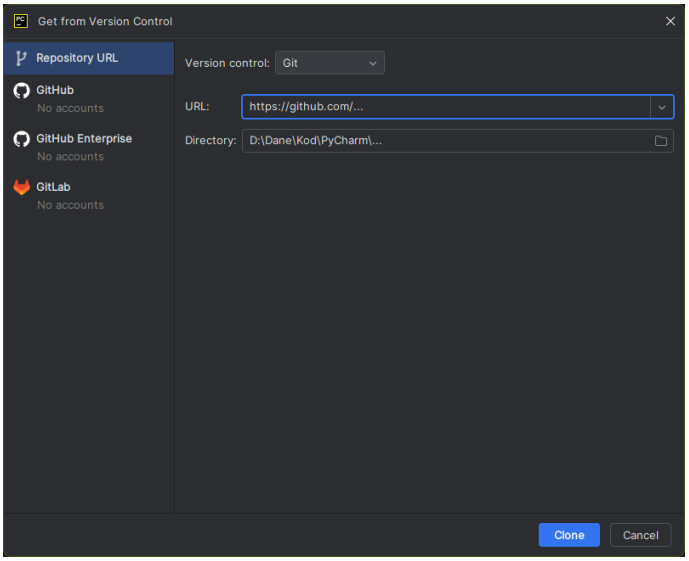


Weryfikacja utworzenia projektu

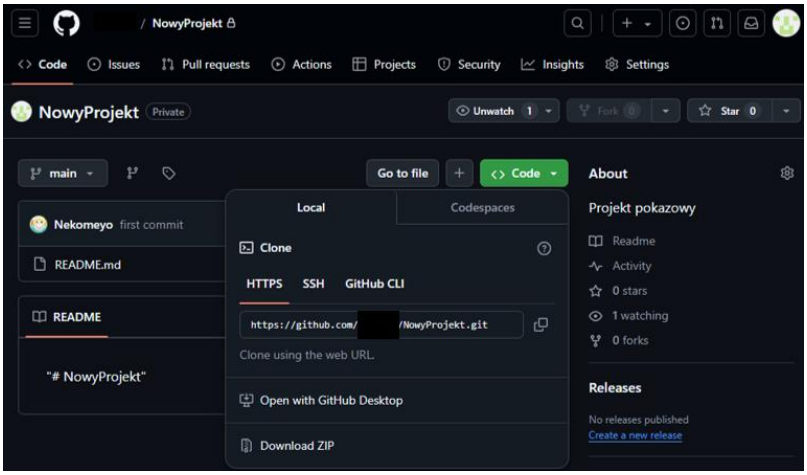


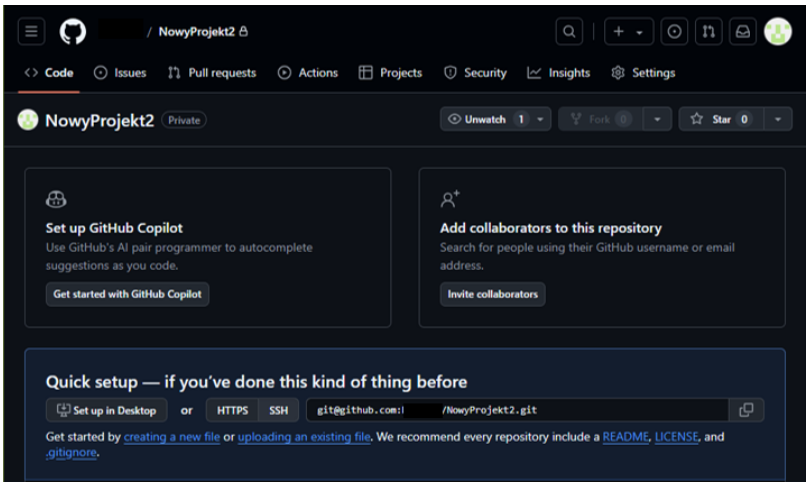
Połączenie z repozytorium GitHub

Aby pobrać cały projekt, który był już wcześniej zamieszczony na GitHub wystarczy wybrać opcję **Get from VCS**, a następnie wkleić URL repozytorium oraz wybrać lokalizację jego zapisania.



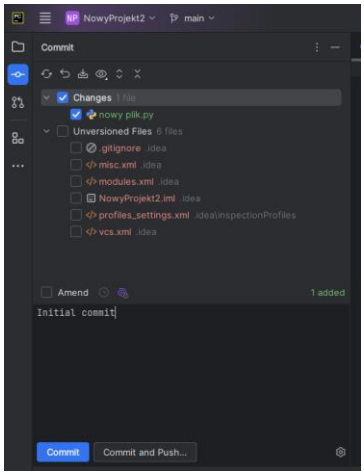
Link do repozytorium, znajduje się na Github:

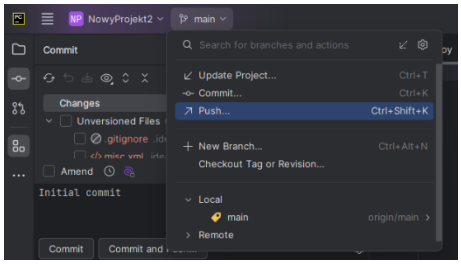


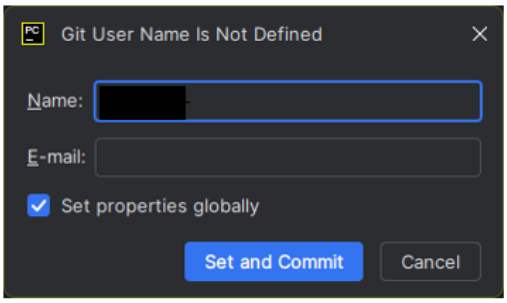


Wprowadzanie zmian do repozytorium

Po zakończeniu każdego etapu prac nad zadaniem należy wybrać opcję **Commit** w celu zatwierdzenia zmian w lokalnym repozytorium. Dodaj komunikat opisujący wprowadzone zmiany i kliknij **Commit.**



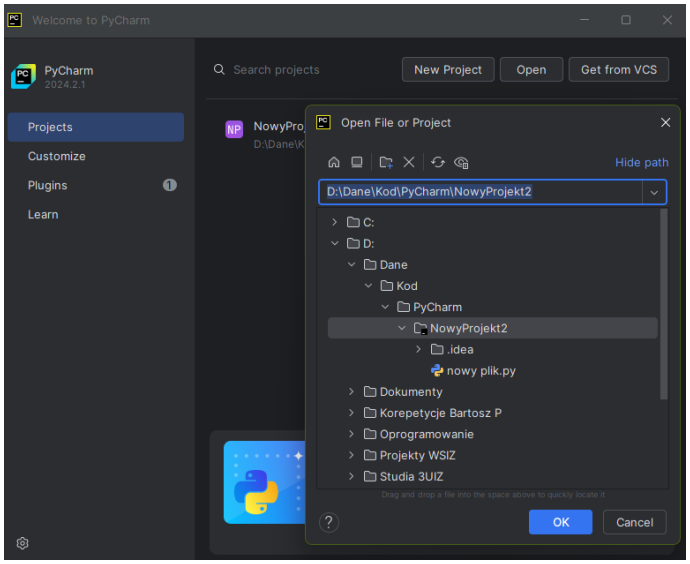


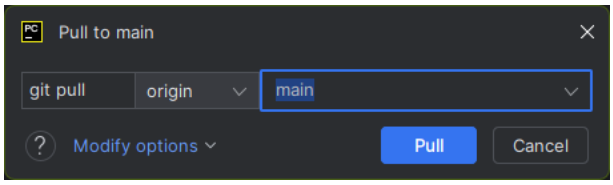


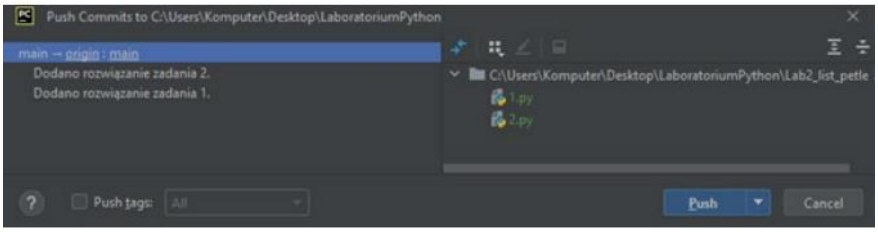
Istnieje możliwość pojawienia się okna dialogowego pytającego o login i e-mail. Należy podać dane zgodne z danymi konta GitHub(!). Na koniec należy przesłać zmiany na serwer **Git -> Push**.

Pobieranie danych z repozytorium

Jeżeli po poprzednich zajęciach zachowało się lokalne repozytorium (np. na dysku F:) w katalogu, którego używałeś, to otwórz je, a następnie pobierz aktualną wersję repozytorium z serwera wybierając w PyCharm opcję Open oraz odpowiedni katalog. W otwartym już projekcie pozostaje użyć **Git -> Pull**.







Składnia języka Python

Składnia jest zbiorem reguł, które określają, jak kod powinien być napisany, aby był poprawny. Kluczowe elementy, z którymi należy się zapoznać aby strukturę składni języka Pythona:

Zmienne: Służą do przechowywania danych, takich jak liczby, tekst czy wartości logiczne

Identyfikator - jest nazwą zmiennej lub funkcji. Nazwy zmiennych i funkcji powinny być unikalne i zgodne z zasadami składni języka.

Operator - jest używany do wykonywania operacji na zmiennych lub funkcjach. Operatory mogą być matematyczne, logiczne lub porównawcze.

I**nstrukcja** - jest używana do wykonywania określonych czynności. Instrukcje mogą być wywoływane przez funkcje lub zmienne.

Blok kodu - jest zbiorem instrukcji wykonywanych jako jedna całość. Bloki kodu są używane do **tworzenia instrukcji warunkowych, pętli i funkcji**.

Instrukcje warunkowe (if-else): Pozwalają na wykonywanie określonych czynności w zależności od spełnienia warunku.

Pętle (for, while): Umożliwiają powtarzanie określonych czynności przez określoną liczbę razy lub do momentu spełnienia określonego warunku.

Funkcje: Pozwalają na grupowanie kodu w celu wielokrotnego wykorzystania i zwiększenia czytelności.

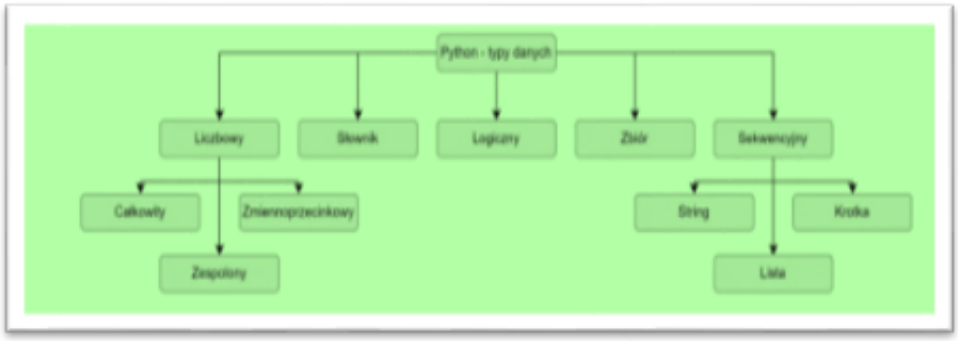
Listy: Umożliwiają przechowywanie wielu elementów w jednej zmiennej.

Słowniki: Pozwalają na przechowywanie par klucz-wartość w jednej zmiennej.

Komentarze: W Pythonie każda linia instrukcji zawierająca symbol # (ang. hash, pl. krzyżyk) oznacza początek komentarza, reszta linii zostaje zignorowana po uruchomieniu programu. Drugim rodzajem komentarzy jest komentarz blokowy, nazywany również wielowierszowy lub wielolinijkowy. Pozwala on na pisanie dowolnych znaków w obrębie wielu linii. Tworzymy go dodając 3x apostrofy bez spacji (lub 3x znaki cudzysłów), a następnie tam gdzie ma się on kończyć, wstawiamy kolejne 3 apostrofy**.** Skrót klawiszowy do automatycznego zakomentowania/odkomentowania zaznaczonego kodu: 'Ctrl'+'/'

Typy danych w języku Python - typ danych określa jakiego "rodzaju" są dane oraz jakie operacje można na nich wykonać. Wbudowane typy danych to takie typy danych, z których można korzystać bez konieczności importowania zewnętrznej biblioteki.
W Pythonie zmienne automatycznie rozpoznają typ danych na podstawie przypisywanej im wartości. Za pomocą konstruktorów typów danych można przypisać do zmiennej typ danych niezależny od typu wartości, którą przypisujemy. np.

```
a = int('5')
print(type(a))
```



Format	Typ danych	Opis

Tekstowy	str	**Typ tekstowy (str)** – reprezentuje sekwencje (ciągi) danych znakowych. Określa się też go jako napisy, łańcuchy znaków.
Liczbowy	int, float, complex	**Typ int** (ang. integer) reprezentuje liczby całkowite dodatnie lub ujemne bez kropki dziesiętnej. Nie mają one limitu dopuszczalnych wartości **Typ float (zmiennoprzecinkowy)** reprezentuje liczby rzeczywiste. Składają się one z części całkowitej i ułamkowej. Separatorem dzielącym część całkowitą od ułamkowej jest w Pythonie kropka. **Liczby zespolone (complex)** – zapisywane są jako suma części rzeczywistej i części urojonej. Część urojona oznaczana jest przez dostawioną na jej końcu literę „j”. Wartości rzeczywista i urojona przechowywane są jako typ float.
Sekwencji	list, tuple, range	**Typ danych lista** – lista to uporządkowana kolekcja elementów. Elementy listy mogą być zmieniane oraz mogą występować duplikaty. **Typ danych krotka** (ang. tuple) – krotka to uporządkowana kolekcja elementów. Elementy krotki nie mogą być zmieniane. Krotka może zawierać duplikaty. **Typ danych** range jest wbudowanym typem danych, który służy do tworzenia sekwencji liczb całkowitych w ustalonego zakresu. Jest to przydatne narzędzie do generowania ciągu liczb, szczególnie w pętlach.
Odwzorowania	dict	**Typ danych dict** – słownik (ang. dictionary) składa się ze zbioru par klucz-wartość. Każda para klucz-wartość odwzorowuje klucz na powiązaną wartość. Klucze muszą być unikalne.
Zestawów	set, frozenset	**Typ danych zbiór** (ang. set) – to nieuporządkowany typ danych kolekcji. Jest on iterowalny, zmienny i nie zawiera zduplikowanych elementów. Zbiór, w przeciwieństwie do listy, posiada zoptymalizowaną, opartą o tablicę haszującą, metodę sprawdzania, czy określony element jest w nim zawarty.

		Typ danych zbiór frozen (ang. frozenset) – to niezmienna wersja obiektu zbiór (set). Elementów w obiekcie frozenset nie można zmieniać.
Logiczny	bool	**Typ danych logiczny** (ang. bool) – umożliwia ocenę dowolnego wyrażenia i uzyskanie odpowiedzi Prawda lub Fałsz.
Binarny	bytes, bytearray,	**Typ bytes** – jest kontenerem do przechowywania bajtów. W przykładzie litera b przed słowem „kot" sygnalizuje, że to jest typ byte a nie string. Typ byte nie jest mutowalny. *zwierze = b"kot"* **Typ bytearray** to zmienna (mutowalna) wersja typu bytes.
Brak	NoneType	Typ NoneType posiada tylko jedną instancję, którą jest None. None jest głównie używany, gdy wartość nie została określona lub nie jest znana.

Operatory arytmetyczne

Operatory arytmetyczne służą do wykonywania wszelkiego rodzaju działań na liczbach takich jak:

Operator	Opis	Przykład
+	dodawanie	2+1 = 3
–	odejmowanie	6-2 = 4
*	mnożenie	3*4 = 12
/	dzielenie zmiennoprzecinkowe	14 / 4 = 3.5
//	dzielenie całkowite	14 // 4 = 3
%	dzielenie modulo (reszta z dzielenia)	16 % 7 = 2 (bo 2*7+2 = 16)
**	potęgowanie	3**4 = 81
+=	zwiększ o	a=2; a+=1 (wynik: a=3)
-=	zmniejsz o	a=6; a-=1 (wynik: a=5)
*=	pomnóż przez	a=5; a*=3 (wynik: a=15)
/=	podziel przez (dzielenie zmiennoprzecinkowe)	a=7; a/=2 (wynik: a=3.5)

//=	podziel przez (dzielenie całkowite)	a=5.0; a//=2 (wynik: a=2.0)

Operatory relacyjne

Operatory relacyjne stosujemy w sytuacje, gdzie jest potrzeba porównania dwóch elementów. Najczęściej używane są w instrukcjach warunkowych i iteracyjnych. Wyróżniamy:

- "<" - mniejszy
- ">" - większy
- "<=" - mniejszy równy
- ">=" - większy równy
- "==" - równy
- "!=" - nie równy

Przypisania wartości

- += - zwiększenie o
- -= - zmniejszenie o
- *= - zwiększenie razy
- /= - zmniejszenie razy

Operacje wejścia i wyjścia

Operacje wejścia i wyjścia (I/O) to sposób w jaki program komunikują się z użytkownikiem, są one kluczowym elementem w każdym programie. Wejście (input) oznacza wszelkie dane, które program otrzymuje od użytkownika lub innego źródła zewnętrznego. Wyjście (output) to dane generowane przez program i przekazywane użytkownikowi lub innemu systemowi. Język Python operacje wejścia i wyjścia realizuje za pomocą wbudowanych funkcji .

Do wyświetlania komunikatów dla użytkownika używamy funkcji print(). Np.

print("Cześć")

Domyślnie print kończy wypisywanie znakiem nowej linii (\n), ale możemy to zmienić, ustawiając end na inny znak lub pusty ciąg znaków.

print("Tekst o nowej linii", end="\n")

print(" - kontynuacja nowej linii")

Jeśli chcemy, aby po wypisaniu tekstu nie było żadnego dodatkowego znaku, możesz ustawić end na pusty ciąg znaków:

print("Tekst bez nowej linii", end="")

print(" - kontynuacja w tej samej linii")

Do pobrania danych od użytkownika możemy użyć funkcji input(). Np.

imie = input("Jak masz na imię? ")

print("Cześć,", imie)

Zadania podstawowe

Zad. 1:

A) Sprawdź w interpreterze typ wyników określonych działań - type(x) i wyjaśnij, co oznaczają poszczególne operatory?

1. 1 + 2
2. 1 + 4.5
3. 3 / 2
4. 4 / 2
5. 3 // 2
6. -3 // 2
7. 11 % 2
8. 2 ** 10
9. 8 ** (1/3)

B) Sprawdź i wyjaśnij działanie następujących instrukcji:

1. int(3.0)
2. float(3)
3. float("3")
4. str(12.4)
5. bool(0)

Zad. 2:

Do zmiennej o nazwie *uczelnia* przypisz zdanie *Studiuję na WSIiZ,* Następnie korzystając z funkcji print() wydrukuj ten tekst do konsoli.

Zad. 3:
Podane są poniższe zmienne:

imię = 'Jan'

wiek = 20

wzrost = 178

Wykorzystując funkcję print() oraz podane zmienne wydrukuj poniższy tekst do konsoli.

Nazywam się Jan i mam 20 lat.

Mój wzrost to 178 cm.

Zad. 4:

Do zmiennej Cena przypisz cenę produktu równą 39.99 PLN oraz do zmiennej Rabat przypisz wartość 0.2 (rabat 20%). Następnie policz cenę tego produktu po zastosowaniu podanego rabatu. Wynik wydrukuj do konsoli. Zwróć uwagę na odpowiednie formatowanie tekstu w funkcji print() tak, aby końcowa cena produktu została wyświetlona tylko do drugiego miejsca po przecinku.

Zad. 5:

Napisz skrypt, który pobiera długości boków prostokąta, a następnie oblicza jego pole i obwód oraz wyświetla wyniki na ekranie.

Zad. 6:

Napisz skrypt, który pobiera od użytkownika drogę pokonaną przez samochód oraz średnie spalanie (w litrach na 100 km) i wyświetli informację o przewidywanym zużyciu paliwa oraz o szacowanych kosztach podróży (cena paliwa 6.5 zł/l).

A) Zmodyfikuj skrypt tak, aby długość przejechanej drogi była generowana losowo (liczba całkowita z zakresu), a użytkownik podawał aktualną cenę paliwa za litr.
B) Zmodyfikuj zadania 6 tak, aby wyświetlanie wyników wykorzystywało f-string.

Podpowiedź:

import random - wykorzystanie w pliku biblioteki pozwalającej na wykorzystywanie
funkcji pseudolosowych los = random.randint(-2, 5) - czytanie do zmiennej los losowej liczby z zakresu <-2, 5>
help(random.randint) - Wyświetlenie pomocy na temat funkcji losującej liczby całkowite
Python umożliwia tworzenie formatowanych łańcuchów tekstowych (stringów), których zawartość zależy od zmiennych, wchodzących w jego skład. Przed tego typu łańcuchem tekstowym wstawiamy literę f, natomiast zmienne do łańcucha wstawiamy w nawiasach klamrowych Na przykład:

```
imie = "Anna"
print(f"Nazywam się {imie}")
```

Zadania dodatkowe

Zad. 7:

Narysuj schemat blokowy algorytmu i napisz program rozwiązywania równania liniowego $ax + b = 0$, gdzie a i b są współczynnikami podawanymi przez użytkownika

Zad. 8:

Narysuj schemat blokowy algorytmu i napisz program rozwiązywania równania kwadratowego $ax^2 + bx + c = 0$, gdzie a, b i c są współczynnikami podawanymi przez użytkownika.

Zad. 9:
Napisz kalkulator, który wyświetli wyniki dodawania, odejmowania, mnożenia, dzielenia i potęgowania 2 liczb podanych przez użytkownika.