KARTA PRACY • KURS PODSTAWOWY PYTHON

stringi.py

Praktyczne operacje na ciągach znaków

Nauczysz się pobierać tekst od użytkownika, sprawdzać jego zawartość, wycinać fragmenty i przekształcać napisy w użyteczne informacje.

PYTHON 3 | 4 lekcje × 45 min | Praca w edytorze kodu | Od podstaw do projektu

TWOJA ŚCIEŻKA

01 · Poznaj napis

Typ `str`, długość, indeksy.

02 · Wycinaj

Fragmenty, łączenie, `in`.

03 · Przekształcaj

Metody tekstowe i czyszczenie.

04 · Zbuduj miniaplikację

Prosty analizator tekstu i sprawdzanie danych wprowadzonych przez użytkownika.

Imię i nazwisko: .....

Klasa / data: .....

Przygotowanie: Python 3, edytor kodu (np. Thonny, VS Code) lub konsola. Wszystkie przykłady można uruchomić bez instalowania dodatkowych bibliotek.

START · 5 MIN

Jak pracować z tym zeszytem?

> **Cel lekcji:** potrafię zapisać tekst w zmiennej, odczytać jego znak lub fragment, zmienić zapis tekstu oraz wykorzystać te operacje w samodzielnym programie.

Plan czterech lekcji

Lekcja	Temat	Proponowany przebieg
1	String, `len()`, indeksy	10 min wprowadzenie · 10 min pokaz · 20 min zadania · 5 min omówienie
2	Wycinki i wyszukiwanie	5 min powtórka · 15 min przykłady · 20 min ćwiczenia · 5 min podsumowanie
3	Metody i porządkowanie danych	5 min powtórka · 15 min pokaz · 20 min ćwiczenia · 5 min sprawdzenie
4	Projekt praktyczny + testy	5 min przypomnienie · 30 min projekt · 5 min testy · 5 min refleksja

Zasady pracy z kodem

1. Przepisz przykład do pliku `.py`, a potem go uruchom.
2. Spróbuj przewidzieć wynik **zanim** zobaczysz go w konsoli.
3. Po każdym zadaniu zmień dane testowe i uruchom program ponownie.
4. Zadania oznaczone **WYZWANIE** wykonaj po ukończeniu podstawy.

Słownik na start

Słowo	Znaczenie
string / str	ciąg znaków, czyli tekst, np. `"Ala"`
indeks	numer pozycji znaku w napisie
wycinek (slice)	fragment napisu wybrany za pomocą `[start:stop]`
metoda	operacja wywołana na obiekcie, np. `tekst.upper()`

> **Uwaga:** Python rozróżnia wielkie i małe litery. Napis `"Kot"` nie jest równy napisowi `"kot"`.

LEKCJA 1 · TEORIA + POKAZ

1. Czym jest string?

String (typ `str`) przechowuje **tekst**: pojedyncze litery, słowa, liczby zapisane jako tekst, spacje oraz inne znaki. W Pythonie napis umieszczamy w apostrofach albo cudzysłowach.

```
miasto = "Lublin"
imie = 'Ola'
kod = "2026"
print(miasto)
print(type(kod))
```

```
Lublin
<class 'str'>
```

> **Zapamiętaj:** `"123"` to tekst, a `123` to liczba całkowita. Nie wykonujemy na nich tych samych operacji bez odpowiedniej konwersji.

Pobieranie tekstu i jego długość

```
imie = input("Podaj imię: ")
print("Witaj, " + imie + "!")
print("Liczba znaków:", len(imie))
```

Funkcja `input()` zwraca tekst (typ `str`). `len()` liczy znaki: litery, cyfry, spacje i znaki interpunkcyjne.

Tekst	`len(tekst)`	Dlaczego?
`"kot"`	3	Trzy litery.
`"kot "`	4	Spacja na końcu też się liczy.
`"zażółć"`	6	Polskie litery są znakami.
`""`	0	To pusty napis.

Znaki specjalne

```
print("Pierwsza linia\nDruga linia")
print("Kolumna 1\tKolumna 2")
print("Cytat: \"Python jest prosty\"")
```

> **Ważne:** `\n` oznacza nową linię, a `\t` tabulator. Nie myl `\n` z dwoma dosłownymi znakami.

LEKCJA 1 · TEORIA + POKAZ

2. Indeksy i odczytywanie znaków

Każdy znak w napisie ma swój indeks. Python zaczyna numerowanie od **zera**.

```
napis = "PYTHON"

znak:     P   Y   T   H   O   N
indeks:   0   1   2   3   4   5
od końca: -6  -5  -4  -3  -2  -1
```

```
napis = "PYTHON"
print(napis[0])   # pierwszy znak
print(napis[2])   # trzeci znak
print(napis[-1])  # ostatni znak
print(napis[-2])  # przedostatni znak
```

```
P
T
N
O
```

Możesz też odczytać ostatnią literę niezależnie od długości napisu:

```
slowo = "informatyka"
print(slowo[len(slowo) - 1])
```

> **Typowy błąd:** `napis[6]` dla `"PYTHON"` powoduje `IndexError` — dostępne indeksy to 0–5. Pusty napis nie ma nawet indeksu 0.

Czy można zmienić pojedynczą literę?

```
slowo = "kot"
# slowo[0] = "p"  # TypeError – tego nie można zrobić
slowo = "p" + slowo[1:]  # tworzymy nowy napis
print(slowo)
```

```
pot
```

> **String jest niemodyfikowalny (immutable).** Możesz odczytywać znaki, ale zmiana zawartości wymaga utworzenia nowego napisu.

LEKCJA 1 · ĆWICZENIA

3. Zadania: od zera do pierwszych znaków

Zadanie 1.1 · Powitanie — PODSTAWA

Pobierz imię użytkownika. Wyświetl napis `Witaj, ...!` i liczbę znaków imienia. Zapisz jako `L1_zad1.py`.

Zadanie 1.2 · Zgadnij wynik — PODSTAWA

Bez uruchamiania kodu podaj wynik obu instrukcji. Następnie sprawdź.

```
tekst = "Ala ma kota"
print(len(tekst))
print(tekst[4])
```

Zadanie 1.3 · Pierwszy i ostatni — PODSTAWA

Pobierz słowo i wypisz pierwszy oraz ostatni znak. Załóż, że użytkownik poda co najmniej jedną literę.

Zadanie 1.4 · Magazyn części — PODSTAWA

Zmienna `kod = "CPU-5070"`. Wypisz: pierwszy znak, znak o indeksie 4 oraz ostatni znak.

Zadanie 1.5 · Porównaj długości — WYZWANIE

Wpisz dwa słowa. Wyświetl, które ma więcej znaków, albo informację, że są równej długości. Użyj `if / elif / else`.

Zadanie 1.6 · Bez błędu dla pustego tekstu — WYZWANIE

Rozbuduj zadanie 1.3: jeżeli użytkownik poda pusty tekst, zamiast pierwszej litery wyświetl `Nie podano słowa`.

> **Minipodsumowanie:** `len()` mierzy długość napisu, `napis[0]` odczytuje pierwszy znak, a `napis[-1]` ostatni.

Szybka kontrola: Czy `len("Ala Ola")` wynosi 6 czy 7? Dlaczego?

LEKCJA 2 · TEORIA + POKAZ

4. Wycinki: wybieranie fragmentu tekstu

Wycinek pozwala pobrać kilka znaków na raz. Ma postać `tekst[start:stop]`. **Początek jest wliczany, koniec nie.**

```
tekst = "INFORMATYKA"
print(tekst[0:4])  # indeksy 0,1,2,3
print(tekst[4:])   # od indeksu 4 do końca
print(tekst[:3])   # pierwsze trzy znaki
print(tekst[-3:])  # trzy ostatnie znaki
```

```
INFO
RMATYKA
INF
YKA
```

Składnia	Znaczenie	Przykład dla `ABCDEF`
`t[1:4]`	Od indeksu 1 do 3	`BCD`
`t[:2]`	Dwa pierwsze znaki	`AB`
`t[3:]`	Od indeksu 3 do końca	`DEF`
`t[-2:]`	Dwa ostatnie znaki	`EF`
`t[::2]`	Co drugi znak	`ACE`
`t[::-1]`	Znaki od końca	`FEDCBA`

> **Przydatne:** `tekst[:3]` nie wywoła błędu, nawet jeżeli tekst ma mniej niż trzy znaki. Odczyt pojedynczego nieistniejącego indeksu już tak.

Łączenie napisów i f-string

```
imie = "Kasia"
klasa = "1TC"
print(imie + " - " + klasa)
print(f"Uczeń: {imie}, klasa: {klasa}")
```

F-string (tekst poprzedzony `f`) pozwala wstawiać zmienne wewnątrz klamer `{}`.

LEKCJA 2 · TEORIA + POKAZ

5. Szukanie tekstu i praca w pętli

Czy tekst zawiera fragment?

Operator `in` daje wynik logiczny `True` lub `False`.

```
adres = "uczen@szkola.pl"
print("@" in adres)
print(".pl" in adres)
print("WWW" in adres)
```

```
True
True
False
```

Możesz to wykorzystać w instrukcji warunkowej:

```
email = input("Podaj e-mail: ")
if "@" in email:
    print("Adres zawiera @")
else:
    print("Brakuje znaku @")
```

To nie jest pełna walidacja adresu e-mail! Sprawdzamy tylko obecność konkretnego znaku.

Przechodzenie po znakach

```
tekst = "KOT"
for znak in tekst:
    print(znak)
```

```
K
O
T
```

Możemy zliczyć wystąpienia wskazanej litery, nawet bez specjalnej metody:

```
slowo = "banan"
licznik = 0
for znak in slowo:
    if znak == "a":
        licznik = licznik + 1
print("Liczba liter a:", licznik)
```

```
Liczba liter a: 2
```

W praktyce: połączenie `if`, `in`, pętli `for` i wycinków wystarczy do zbudowania wielu prostych narzędzi tekstowych.

LEKCJA 2 · ĆWICZENIA

6. Zadania: wycinanie i wykrywanie

Zadanie 2.1 · Kod produktu PODSTAWA

Dla `kod = "LAP-2026-ASUS"` wypisz: **LAP**, **2026** i **ASUS**. Użyj wyłącznie wycinków.

Zadanie 2.2 · Inicjały PODSTAWA

Pobierz imię i nazwisko. Wypisz pierwsze litery, np. `Jan Kowalski → JK`. Załóż, że oba pola są niepuste.

Zadanie 2.3 · Przywitaj po swojemu PODSTAWA

Wpisz imię i miasto. Wyświetl jedno zdanie wykorzystujące f-string: `Ola mieszka w Lublinie.`

Zadanie 2.4 · Sprawdź obecność PODSTAWA

Pobierz zdanie i fragment tekstu. Wyświetl informację, czy fragment występuje w zdaniu. Użyj `in`.

Zadanie 2.5 · Ile jest liter a? PODSTAWA

Pobierz słowo. Używając pętli `for`, policz małe litery `a`. Możesz oprzeć się na przykładzie z poprzedniej strony.

Zadanie 2.6 · Tekst od tyłu WYZWANIE

Pobierz dowolny napis i wyświetl go w odwrotnej kolejności, wykorzystując wycinek `[::-1]`.

Zadanie 2.7 · Numer serii WYZWANIE

Dla `numer = "ZSP-1TC-2026"` pobierz fragment `1TC`. Następnie przetestuj program na innym numerze o tym samym układzie.

Pytanie kontrolne: Co wyświetli `print("ABCDE"[1:4])`? Czy znak o indeksie 4 znajdzie się w wyniku?

LEKCJA 3 · TEORIA + POKAZ

7. Metody: przekształcanie tekstu

Metoda jest operacją wykonywaną na napisie. Zapisujemy ją po kropce, na przykład `tekst.upper()`. Większość pokazanych metod **zwraca nowy napis** i nie zmienia oryginału.

```
tekst = "  Python jest SUPER!  "
print(tekst.lower())
print(tekst.upper())
print(tekst.strip())
print(tekst.replace("SUPER", "ciekawy"))
```

Metoda	Co robi?	Przykład → wynik
`lower()`	Małe litery	`"Kot" → "kot"`
`upper()`	Wielkie litery	`"kot" → "KOT"`
`capitalize()`	Wielka pierwsza litera, pozostałe małe	`"lUBLIN" → "Lublin"`
`strip()`	Usuwa spacje i inne białe znaki z obu końców	`" Ola " → "Ola"`
`replace(a, b)`	Zastępuje fragment `a` tekstem `b`	`"kot" → "lot"` przy `replace("k", "l")`
`count(x)`	Liczy wystąpienia fragmentu `x`	`"banan".count("a") → 2`
`find(x)`	Indeks pierwszego wystąpienia, albo `-1`	`"kotek".find("te") → 2`

Połącz metody: `input("Imię: ").strip().capitalize()` pozwala od razu poprawić wpisane imię, np. `" oLA "` na `"Ola"`.

LEKCJA 3 · TEORIA + POKAZ

8. Dzielenie tekstu i sprawdzanie danych

split() – podziel napis na elementy

```
dane = "monitor;mysz;klawiatura"
lista = dane.split(";")
print(lista)
print(lista[1])
```

```
['monitor', 'mysz', 'klawiatura']
mysz
```

`split(";")` dzieli tekst w miejscach średników. Wynikiem jest **lista**, z której wybierasz element przez indeks. `split()` bez argumentu dzieli po białych znakach (np. spacjach).

join() – połącz elementy listy

```
czesci = ["CPU", "5070", "GPU"]
print("-".join(czesci))
```

```
CPU-5070-GPU
```

Metody sprawdzające – wynik True / False

Metoda	Wynik True, gdy...
`isalpha()`	Wszystkie znaki są literami (i napis nie jest pusty).
`isdigit()`	Wszystkie znaki są cyframi (i napis nie jest pusty).
`startswith(x)`	Tekst zaczyna się od fragmentu `x`.
`endswith(x)`	Tekst kończy się fragmentem `x`.

```
pin = input("Podaj 4-cyfrowy PIN: ").strip()
if len(pin) == 4 and pin.isdigit():
    print("Format PIN-u poprawny")
else:
    print("Niepoprawny format")
```

> **Uwaga bezpieczeństwa:** to przykład sprawdzania *formatu*, a nie prawdziwego uwierzytelniania. Nie zapisuj ani nie wyświetlaj rzeczywistych kodów PIN innych osób.

LEKCJA 3 · ĆWICZENIA

9. Zadania: porządkuj dane użytkownika

Zadanie 3.1 · Wyrównaj imię

PODSTAWA

Pobierz imię, które może mieć spacje na początku i końcu oraz błędne wielkie litery. Wyświetl je po użyciu `strip()` i `capitalize()`.

Zadanie 3.2 · Policz literę

PODSTAWA

Pobierz zdanie i policz wystąpienia `a` bez rozróżniania wielkości liter. Użyj `lower()` i `count()`.

Zadanie 3.3 · Zamiana słowa

PODSTAWA

Pobierz zdanie. Zamień słowo `komputer` na `laptop` za pomocą `replace()`. Zwróć uwagę na wielkość liter.

Zadanie 3.4 · Lista zakupów IT

PODSTAWA

Dla napisu `"RAM,SSD,CPU"` użyj `split(",")` i wypisz każdy element listy w osobnym wierszu, np. pętlą `for`.

Zadanie 3.5 · Poprawność numeru

PODSTAWA

Pobierz identyfikator. Zaakceptuj go tylko wtedy, gdy zawiera dokładnie 6 cyfr. Użyj `len()`, `isdigit()` i `and`.

Zadanie 3.6 · Wyszukiwarka

WYZWANIE

Pobierz zdanie i słowo. Wypisz pozycję pierwszego wystąpienia z `find()`. Gdy wynik wynosi `-1`, pokaż `Nie znaleziono`.

Zadanie 3.7 · Nazwa pliku

WYZWANIE

Pobierz nazwę pliku i sprawdź, czy po `strip().lower()` kończy się na `.py`.

Porównaj: `" Ola ".strip()` i `" Ola ".replace(" ", "")`. Czy dla tekstu `"Ala ma kota"` działają tak samo?

LEKCJA 4 · PROJEKT

10. Projekt: analizator tekstu

> **Misja:** zaprojektuj prosty program, który po wpisaniu tekstu pokaże podstawowe informacje o nim. Projekt ma działać w zwykłej konsoli Pythona.

Minimalne wymagania – wykonaj wszystkie

1. Pobierz od użytkownika dowolny tekst przez `input()`.
2. Wyświetl tekst bez spacji na początku i końcu (`strip()`).
3. Podaj liczbę znaków **po usunięciu spacji z brzegów**.
4. Wyświetl tekst zapisany wielkimi literami.
5. Policz wystąpienia litery `a` bez rozróżniania wielkości liter.
6. Sprawdź, czy w tekście pojawia się fragment `python` (niezależnie od wielkości liter).

Przykładowy test

Użytkownik wpisuje: `Ala lubi Python!`

```
Tekst po oczyszczeniu: Ala lubi Python!
Liczba znaków: 16
DUŻE LITERY: ALA LUBI PYTHON!
Liczba liter a: 2
Czy zawiera "python"? Tak
```

W tym przykładzie `len("Ala lubi Python!")` to 16 — spacje wewnątrz zdania są liczone.

Szkielet kodu do uzupełnienia

```
tekst = input("Wpisz tekst: ")
czysty = tekst.strip()
print("Tekst po oczyszczeniu:", czysty)

# 1. Dopisz liczbę znaków czystego tekstu.
# 2. Wyświetl napis wielkimi literami.
# 3. Policz litery a po zamianie na małe.
# 4. Użyj if i operatora in do wykrycia "python".
```

> **Kryterium sukcesu:** program działa dla pustego napisu, pojedynczego słowa i zdania ze spacjami; nie kończy się błędem.

LEKCJA 4 · TESTY I ROZBUDOWA

11. Testowanie – sprawdź, zanim oddasz

Programista nie uruchamia programu tylko raz. Sprawdź poniższe przypadki i wpisz do tabeli: **OK** albo **DO POPRAWY**.

Test	Co ma się wydarzyć?	Wynik
`Ala`	Długość 3, liczba liter a = 2, brak „python”.	.....
`Python`	Długość 6, wykryty „python”.	.....
`PYTHON`	Wykryty „python” mimo wielkich liter.	.....
`bbb`	Liczba liter a = 0.	.....
`[pusty tekst]`	Długość 0, bez błędu.	.....

Rozbudowa dla chętnych

Zadanie 4.1 · Pierwszy i ostatni znak

WYZWANIE

Jeżeli oczyszczony tekst nie jest pusty, wypisz pierwszy i ostatni znak; jeśli jest pusty, pokaż komunikat. Wykorzystaj `if len(czysty) > 0`.

Zadanie 4.2 · Liczba słów

WYZWANIE

Użyj `split()` i `len()`, aby policzyć słowa w tekście. Załóż, że słowa oddzielają białe znaki.

Zadanie 4.3 · Palindrom

WYZWANIE

Sprawdź, czy słowo czyta się tak samo od początku i od końca, np. `kajak`. Najpierw zastosuj `strip().lower()`, potem `[::-1]`.

> **Ograniczenie:** zadanie 4.3 w tym wariancie jest przeznaczone do pojedynczych słów. Aby badać całe zdania, trzeba dodatkowo obsłużyć spacje i znaki interpunkcyjne.

Oddawanie pracy

Zapisz plik jako `L4_analizator_tekstowy.py`. Na górze dodaj komentarz z imieniem, nazwiskiem i klasą. Program powinien uruchamiać się samodzielnie i mieć czytelne komunikaty.

LEKCJA 4 · DRUGI WARIANT

12. Alternatywny projekt: kontrola danych

> **Wybór:** jeżeli skończysz analizator wcześniej, przygotuj walidator krótkiego identyfikatora ucznia do **ćwiczeń na danych fikcyjnych**.

Przyjmij format identyfikatora `TC-1234`: dwie wielkie litery `TC`, myślnik i cztery cyfry.

Warunki poprawności

1. Po usunięciu spacji z brzegów napis ma długość 7.
2. Zaczyna się od `TC-` (wielkie litery).
3. Ostatnie cztery znaki to cyfry.

Podpowiedź – kod startowy

```
identyfikator = input("Podaj identyfikator: ")
identyfikator = identyfikator.strip().upper()

poczatek = identyfikator[:3]
numer = identyfikator[3:]

if len(identyfikator) == 7 and poczatek == "TC-" and numer.isdigit():
    print("Poprawny format")
else:
    print("Błędny format")
```

Przetestuj

Wpis	Oczekiwany rezultat
`TC-1234`	Poprawny format
`tc-4321`	Poprawny format (po oczyszczeniu)
`TC-12A4`	Błędny format
`TI-1234`	Błędny format
`TC-123`	Błędny format

Dlaczego to działa? Operator `and` oznacza, że *wszystkie* warunki muszą być spełnione. Pusta część numeryczna nie zostanie uznana za cyfry przez `isdigit()`.

LEKCJA 4 · POWTÓRZENIE

13. Ściąga – najważniejsze polecenia

Chcę…	Używam	Przykład / wynik
utworzyć napis	`str`	`tekst = "kot"`
sprawdzić długość	`len(t)`	`len("Ala") → 3`
pierwszy / ostatni znak	`t[0]`, `t[-1]`	`"kot"[-1] → "t"`
wyciąć fragment	`t[a:b]`	`"Python"[:2] → "Py"`
odwrócić kolejność	`t[::-1]`	`"kot" → "tok"`
połączyć napisy	`+` lub f-string	`f"Cześć {imie}"`
sprawdzić fragment	`x in t`	`"a" in "Ala" → True`
zmienić wielkość liter	`lower()`, `upper()`	`"Kot".lower() → "kot"`
usunąć spacje z brzegów	`strip()`	`" kot ".strip() → "kot"`
zastąpić fragment	`replace(a,b)`	`"kot".replace("k","l")`
policzyć wystąpienia	`count(x)`	`"banan".count("a") → 2`
znaleźć pozycję	`find(x)`	`"abc".find("z") → -1`
podzielić / połączyć listę	`split()`, `join()`	`"a,b".split(",")`
sprawdzić cyfry	`isdigit()`	`"123".isdigit() → True`
sprawdzić początek / koniec	`startswith()`, `endswith()`	`"test.py".endswith(".py")`

Trzy pułapki: indeksy zaczynają się od 0; w wycinku `[a:b]` koniec nie należy do wyniku; `strip()` nie usuwa spacji ze środka zdania.

PODSUMOWANIE · SPRAWDŹ SIĘ

14. Miniquiz i samoocena

Bez uruchamiania kodu wpisz przewidywany wynik. Dopiero potem sprawdź swoje odpowiedzi w Pythonie.

Zadanie Q1 · Długość PODSTAWA

Co wyświetli `print(len("A B"))`?

Zadanie Q2 · Indeksy PODSTAWA

Co wyświetli `print("Pies"[-1])`?

Zadanie Q3 · Wycinek PODSTAWA

Co wyświetli `print("KOMPUTER"[1:4])`?

Zadanie Q4 · Metody PODSTAWA

Co wyświetli `print(" JaN ".strip().lower())`?

Zadanie Q5 · Sprawdzanie PODSTAWA

Jaką wartość zwróci `"2026".isdigit()`?

Zadanie Q6 · Błąd czy nie? PODSTAWA

Dlaczego kod `tekst = "kot"; tekst[0] = "p"` jest niepoprawny?

Moja samoocena

- ☐ Potrafię utworzyć zmienną tekstową i użyć `input()`.
- ☐ Rozumiem działanie `len()` i indeksów.
- ☐ Potrafię wyciąć fragment tekstu.
- ☐ Umiem stosować `strip()`, `lower()` i `replace()`.
- ☐ Potrafię użyć tekstu w warunku `if`.
- ☐ Umiem przetestować program na kilku danych.

> **Na koniec:** Zaznacz jedną operację, którą już dobrze rozumiesz, i jedną, którą chcesz jeszcze przećwiczyć.