

□ Tutorial □

Pętle w Python

Od podstaw do praktyki

5 prostych zadań z kompletnymi wyjaśnieniami

Spis treści

1. Podstawy pętli
2. Pętla for
3. Pętla while
4. Rozwiązywanie zadań krok po kroku
 - Zadanie 1: Wypisz liczby od 1 do 10
 - Zadanie 2: Suma liczb od 1 do 100
 - Zadanie 3: Imiona z listy
 - Zadanie 4: Silnia liczby 5
 - Zadanie 5: Zgadywanka
5. Podsumowanie i najczęstsze błędy
6. Ćwiczenia dodatkowe

1. Podstawy pętli

Co to jest pętla?

Pętla to sposób na powtarzanie tego samego kodu wiele razy. Zamiast pisać ten sam kod wielokrotnie, używamy pętli.

Bez pętli:

```
print("Cześć")  
print("Cześć")  
print("Cześć")
```

Z pętlą:

```
for i in range(3):  
    print("Cześć")
```

2. Pętla for

Podstawowa składnia:

```
for zmienna in sekwencja:  
    # kod do wykonania  
    # wcięcie jest WAŻNE!
```

Przykład 1: Prosta pętla

```
for i in range(5):  
    print(i)
```

Wynik: 0, 1, 2, 3, 4

WAŻNE: range(5) daje liczby od 0 do 4 (nie 5!)

Przykład 2: range() z różnymi parametrami

```
# range(start, stop)  
for i in range(1, 6): # od 1 do 5  
    print(i)  
  
# range(start, stop, step)  
for i in range(0, 10, 2): # co drugi  
    print(i) # 0, 2, 4, 6, 8
```

Przykład 3: Iterowanie po liście

```
owoce = ["jabłko", "banan", "gruszka"]  
  
for owoc in owoce:  
    print(owoc)
```

Przykład 4: enumerate() - lista z numerami

```
owoce = ["jabłko", "banan", "gruszka"]  
  
for numer, owoc in enumerate(owoce):  
    print(f"{numer}. {owoc}")  
  
# Wynik: 0. jabłko, 1. banan, 2. gruszka
```

TIP: Użyj enumerate(owoce, start=1) żeby zacząć od 1!

3. Pętla while

Podstawowa składnia:

```
while warunek:  
    # kod wykonywany dopóki warunek = True
```

Przykład 1: Prosta pętla while

```
licznik = 0  
while licznik < 5:  
    print(licznik)  
    licznik += 1 # WAŻNE: zwiększ licznik!
```

Przykład 2: Pętla nieskończona z break

```
while True:  
    odpowiedz = input("Kontynuować? (tak/nie): ")  
    if odpowiedz == "nie":  
        break # wyjście z pętli  
    print("Kontynuujemy!")
```

4. Rozwiązywanie zadań krok po kroku

ZADANIE 1: Wypisz liczby od 1 do 10

Krok 1: Zrozum zadanie

Musimy wypisać: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 (każda liczba w nowej linii)

Krok 2: Planowanie

Użyjemy pętli for. Potrzebujemy liczb od 1 do 10. range(1, 11) da nam to, czego potrzebujemy.

Krok 3: Kod

```
for i in range(1, 11):  
    print(i)
```

Krok 4: Jak to działa?

```
# i = 1 → print(1)  
# i = 2 → print(2)  
# ...  
# i = 10 → print(10)
```

ZADANIE 2: Suma liczb od 1 do 100

Krok 1: Zrozum zadanie

Musimy dodać: $1 + 2 + 3 + \dots + 100$ i wypisać wynik.

Krok 2: Planowanie

Potrzebujemy zmiennej do przechowywania sumy. W pętli dodajemy każdą kolejną liczbę. Na końcu wypisujemy wynik.

Krok 3: Kod krok po kroku

```
# Krok 1: Stwórz zmienną na sumę
suma = 0

# Krok 2: Przejdź przez liczby od 1 do 100
for i in range(1, 101):
    suma += i # To samo co: suma = suma + i

# Krok 3: Wypisz wynik
print(f"Suma liczb od 1 do 100 wynosi: {suma}")
```

Krok 4: Jak to działa?

```
# Początek: suma = 0
# i = 1: suma = 0 + 1 = 1
# i = 2: suma = 1 + 2 = 3
# i = 3: suma = 3 + 3 = 6
# i = 4: suma = 6 + 4 = 10
# ...
# i = 100: suma = ... + 100 = 5050
```

Typowe błędy:

```
# BŁĄD 1: Zapomniano zainicjować sumy
for i in range(1, 101):
    suma += i # ERROR! suma nie istnieje

# DOBRZE:
suma = 0
for i in range(1, 101):
    suma += i

# BŁĄD 2: Zły zakres
for i in range(1, 100): # Braknie 100!
    suma += i

# DOBRZE:
for i in range(1, 101): # 101 nie jest uwzględnione
    suma += i
```

ZADANIE 3: Imiona z listy

Cel:

Wypisać każde imię z numerem porządkowym: 1. Anna, 2. Bartek, 3. Celina, 4. Dawid

Sposób 1: enumerate() - NAJLEPSZY

```
imiona = ["Anna", "Bartek", "Celina", "Dawid"]
for numer, imie in enumerate(imiona, start=1):
    print(f"{numer}. {imie}")
```

Sposób 2: range() i indeksy

```
imiona = ["Anna", "Bartek", "Celina", "Dawid"]
for i in range(len(imiona)):
    print(f"{i+1}. {imiona[i]}")
```

Sposób 3: Ręczny licznik

```
imiona = ["Anna", "Bartek", "Celina", "Dawid"]
numer = 1
for imie in imiona:
    print(f"{numer}. {imie}")
    numer += 1
```

Który sposób wybrać? Sposób 1 jest najbardziej pythonowy i czytelny!

ZADANIE 4: Silnia liczby 5

Krok 1: Zrozum zadanie

Silnia: $5! = 1 \times 2 \times 3 \times 4 \times 5 = 120$. Mnożymy wszystkie liczby od 1 do 5.

Krok 2: Planowanie

Potrzebujemy zmiennej na wynik (zaczynamy od 1, nie 0!). W pętli mnożymy przez kolejne liczby.

Krok 3: Kod

```
n = 5
silnia = 1

for i in range(1, n + 1):
    silnia *= i # To samo co: silnia = silnia * i

print(f"Silnia z {n} wynosi: {silnia}")
```

Krok 4: Jak to działa?

```
# Początek: silnia = 1
# i = 1: silnia = 1 * 1 = 1
# i = 2: silnia = 1 * 2 = 2
# i = 3: silnia = 2 * 3 = 6
# i = 4: silnia = 6 * 4 = 24
# i = 5: silnia = 24 * 5 = 120
```

UWAGA: Dlaczego zaczynamy od 1?

```
# BŁĄD: Start od 0
silnia = 0
for i in range(1, 6):
    silnia *= i # 0 * cokolwiek = 0!

# DOBRZE: Start od 1
silnia = 1
for i in range(1, 6):
    silnia *= i
```

ZADANIE 5: Zgadywanka

Krok 1: Zrozum zadanie

Komputer ma sekretną liczbę. Użytkownik zgaduje. Program podpowiada: za mało / za dużo. Powtarzamy aż do odgadnięcia.

Krok 2: Planowanie

Użyjemy pętli while. Potrzebujemy porównać zgadniętą liczbę z sekretem. Musimy przerwać pętlę po odgadnięciu.

Krok 3: Kod podstawowy

```
sekret = 7
print("Zgadnij liczbę od 1 do 10!")
while True:
    zgadnieta = int(input("Twoja liczba: "))
    if zgadnieta < sekret:
        print("Za mało! Spróbuj ponownie.")
    elif zgadnieta > sekret:
        print("Za dużo! Spróbuj ponownie.")
    else:
        print("Brawo! Odgadłeś liczbę!")
        break # Wyjście z pętli
```

Wersja z licznikiem prób:

```
sekret = 7
proby = 0
while True:
    proby += 1
    zgadnieta = int(input(f"Próba #{proby}: "))
    if zgadnieta == sekret:
        print(f"Brawo! Odgadłeś za {proby} razem!")
        break
    elif zgadnieta < sekret:
        print("Za mało!")
    else:
        print("Za dużo!")
```

Wersja z losową liczbą:

```
import random
sekret = random.randint(1, 10)
while True:
    zgadnieta = int(input("Zgadnij (1-10): "))
    if zgadnieta == sekret:
        print("Brawo!")
        break
    elif zgadnieta < sekret:
        print("Za mało!")
    else:
        print("Za dużo!")
```

5. Podsumowanie - Najważniejsze wskazówki

Pętla for - używaj gdy:

- Wiesz ile razy coś powtórzyć
- Przetwarzasz listę/kolekcję
- Liczysz od X do Y

Pętla while - używaj gdy:

- Nie wiesz ile razy powtórzyć
- Czekasz na konkretny warunek
- Robisz grę lub program interaktywny

Najczęstsze błędy:

BŁĄD 1: Brak wcięcia

```
# ŹLE:
for i in range(5):
print(i) # ERROR!

# DOBRZE:
for i in range(5):
    print(i)
```

BŁĄD 2: Zły zakres

```
# ŹLE:
for i in range(10): # 0-9
    print(i) # Braknie 10!

# DOBRZE:
for i in range(11): # 0-10
```

BŁĄD 3: Niezainicjowana zmienna

```
# ŹLE:
for i in range(5):
    suma += i # ERROR! suma nie istnieje

# DOBRZE:
suma = 0
for i in range(5):
    suma += i
```

BŁĄD 4: Nieskończona pętla while

```
# ŹLE:
i = 0
while i < 5:
    print(i)
    # Brak i += 1 - pętla nigdy się nie skończy!

# DOBRZE:
i = 0
while i < 5:
    print(i)
    i += 1
```

6. Ćwiczenia dodatkowe

Ćwiczenie 1: Liczby parzyste

Wypisz wszystkie liczby parzyste od 2 do 20.

Ćwiczenie 2: Tabliczka mnożenia

Wypisz tabliczkę mnożenia dla liczby 7 (7×1 , 7×2 , ... 7×10).

Ćwiczenie 3: Odwrotnie

Wypisz liczby od 10 do 1 (odwrotna kolejność).

Ćwiczenie 4: Suma liczb parzystych

Oblicz sumę wszystkich liczb parzystych od 1 do 50.

Ćwiczenie 5: Gwiazki

Wypisz wzór: * / ** / *** / **** / *****

Powodzenia w nauce!

Po opanowaniu tych zadań możesz przejść do zagnieżdżonych pętli, list comprehensions, pętli z funkcjami i debugowania z pętlami.