

Wstęp do programowania

INP003203L

rok akademicki 2021/22

semestr zimowy

Laboratorium 8

Karol Tarnowski

karol.tarnowski@pwr.edu.pl

L-1 p. 220



Plan prezentacji

- Porównania ciągów znakowych
- Znaki jako liczby
- Dane logiczne



Porównywanie ciągów znakowych

example_if_07.py



File Edit Format Run Options Window Help

```
#Program pokazuje porównanie ciągów tekstowych.
```

```
def main():
```

```
    name1 = 'Maria'
```

```
    name2 = 'Marek'
```

```
    # Operator porównania sprawdza, czy dwa ciągi znaków  
    # są sobie równe - czy są takie same.
```

```
    # W zależności od wyniku porównania instrukcja warunkowa  
    # wybierze jedną z dwóch ścieżek.
```

```
    if name1 == name2:
```

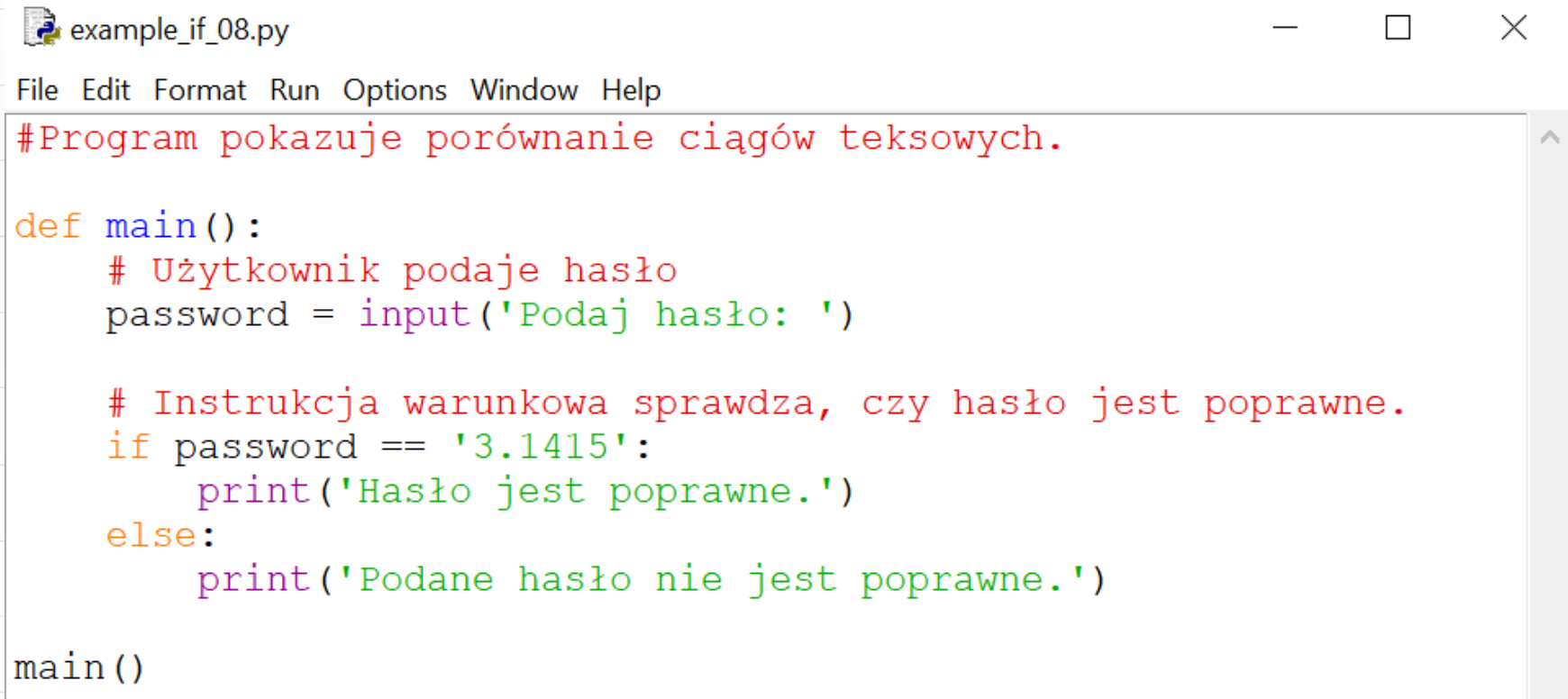
```
        print('Imiona są takie same')
```

```
    else:
```

```
        print('Imiona nie są takie same')
```

```
main()
```

Porównywanie ciągów znakowych



```
example_if_08.py
File Edit Format Run Options Window Help
#Program pokazuje porównanie ciągów tekstowych.

def main():
    # Użytkownik podaje hasło
    password = input('Podaj hasło: ')

    # Instrukcja warunkowa sprawdza, czy hasło jest poprawne.
    if password == '3.1415':
        print('Hasło jest poprawne.')
    else:
        print('Podane hasło nie jest poprawne.')

main()
```

Porównywanie ciągów znakowych

- Operator porównania działa również dla ciągów tekstowych



Znaki jako liczby

Symbole znakowe są przechowywane w pamięci komputera jako liczby.

Najbardziej podstawowy system kodowania to system ASCII (American Standard Code for Information Interchange)

- literom od „A” do „Z” odpowiadają liczby od 65 do 90,
- literom od „a” do „z” odpowiadają liczby od 97 do 122,
- cyfrom od „0” do „9” odpowiadają liczby od 48 do 57,
- znakowi spacji odpowiada liczba 32,
- znakowi nowej linii odpowiada liczba 10.

Kody ASCII nie obejmują polskich znaków diakrytycznych.



Znaki jako liczby

- Rozszerzeniem zestawu znaków ASCII na znaki używane w różnych językach jest zestaw znaków Unicode
- Do zapisywania znaków Unicode wykorzystuje się różne kodowania
- Popularnym kodowaniem jest kodowanie UTF-8 (Unicode Transformation Format), gdzie 8 oznacza, że do kodowania wykorzystywane są liczby 8-bitowe



Znaki jako liczby

- Funkcja `ord()` dla podanego znaku zwraca jego kod liczbowy
- Funkcja `chr()` dla podanego kodu liczbowego zwraca odpowiadający mu znak

Porównywanie ciągów znakowych

example_if_09.py

File Edit Format Run Options Window Help

```
#Program pokazuje porównanie ciągów tekstowych.
```

```
def main():
```

```
    name1 = 'Maria'
```

```
    name2 = 'Marek'
```

```
# Operator większości sprawdza, czy ciąg pierwszy jest  
# większy niż drugi (w porządku alfabetycznym).
```

```
if name1 > name2:
```

```
    print('Imię: ', name1, ' ', sep = ' ', end = '')
```

```
    print(' jest większe niż imię: ', name2, '.', sep = '')
```

```
#   M   a   r   i   a
```

```
#  77  97 114 105  97
```

```
#   M   a   r   e   k
```

```
#  77  97 114 101 107
```

```
# M==M a==a r==r i>e
```

```
main()
```

Porównywanie ciągów znakowych

- Porównywanie ciągów tekstowych sprowadza się do porównywania kodów kolejnych znaków ciągu



Logiczny typ danych

- Dana typu logicznego może przyjmować jedną z dwóch wartości: prawda (**True**) lub fałsz (**False**).

hungry = True

sleepy = False

zadanie 2

Logiczny typ danych

Przykład

example_bool_01.py

File Edit Format Run Options Window Help

```
#Program pokazuje sprawdzanie parzystości liczby.  
#Parzystość liczby jest związana z podzielnością przez 2.
```

```
def main():
```

```
    x = 14
```

```
    # Jeśli liczba dzieli się przez 2 bez reszty  
    # (reszta z dzielenia przez 2 równa się 0),
```

```
    if x % 2 == 0:
```

```
        #to liczba jest parzysta
```

```
        print('Liczba', x, 'jest parzysta')
```

```
    else:
```

```
        #w p. p. jest nieparzysta
```

```
        print('Liczba', x, 'jest nieparzysta')
```

```
main()
```

Logiczny typ danych

Przykład

```
example_bool_02.py
File Edit Format Run Options Window Help
#Program pokazuje sprawdzanie parzystosc liczby.
#Wynik sprawdzenia (wartosc logiczna)
#zapisywany jest w zmiennej even.

def main():
    x = 14

    # Zmienna even zapamieta, czy liczba jest parzysta.
    even = x % 2 == 0

    if even:
        #jest parzysta
        print('Liczba', x, 'jest parzysta')
    else:
        #w p. p. jest nieparzysta
        print('Liczba', x, 'jest nieparzysta')

    # Drugi raz wykorzystujemy obliczona zmienna.
    if even:
        print('Ciagle parzysta')

main()
```



Logiczny typ danych

- Wynik operacji logicznej można przechowywać w zmiennej typu logicznego

Logiczny typ danych

Przykład

example_bool_03.py

File Edit Format Run Options Window Help

```
#Program pokazuje sprawdzanie parzystości liczby.  
#Do sprawdzania parzystości wykorzystywana jest  
#funkcja is_even() zwracająca wartość logiczną.  
  
def main():  
    x = 14  
  
    #wywołanie funkcji is_even() w warunku instrukcji if  
    if is_even(x):  
        print('Liczba', x, 'jest parzysta')  
    else:  
        print('Liczba', x, 'jest nieparzysta')  
  
#I: liczba całkowita  
#P: sprawdzane jest, czy reszta z dzielenia przez 2  
#   równa się 0  
#O: wartość logiczna:  
#   True (jeśli liczba jest parzysta)  
#   False (w p. p.)  
def is_even(x):  
    return x % 2 == 0  
  
main()
```



Logiczny typ danych

- Dane typu logicznego mogą być również zwracane przez funkcje



Absolutne minimum

- W jaki sposób porównywane są łańcuchy znakowe?
- Znaki są pamiętane (kodowane) jako liczby
- Jakie wartości może przyjmować logiczny typ danych?