Print Formatting

Python umożliwia kilka sposobów manipulacji łańcuchami znaków (str) i uwzględnianie zmiennych w wyświetlanych napisach.

Konkatenacja stringów

Operator `+`

Operator + skleja ze sobą stringi.

```
imie = 'Hori Kamatani'
print('My name ' + imie + '!')
```

Można go użyć do wyświetlania zmiennych liczbowych, ale nie jest to najlepsze rozwiązanie.

```
imie = 'Hori Kamatani'
wiek = 35
print('My name is ' + imie + ' and I am ' +str(wiek) + ' years old !')
```

Wykorzystanie parametrów funkcji `print()`

Do łączenia ciągów znakowych, które mają zostać wyświetlone dla użytkownika, można wykorzystać bezpośrednio właściwość funkcji `print`, która może przyjąć wiele argumentów, które następnie ze sobą połączy.

```
imie = 'Hori Kamatani'
print('My name', imie, '!')
```

Można tutaj podać jako argumenty zarówno ciągi znaków jak i wartości liczbowe. Ciągi znaków będą od siebie odseparowane ciągiem znaków przekazanym przez argument `sep`, domyślnie jest to spacja `' '`.

Interpolacja zmiennych

Operator: `%s`, `%d`, `%f`

Używanie tych operatorów przypomina używanie funkcji `printf`, znanej między innymi z C++. W tekście stringa wstawiamy odpowiedni operator: `%s` dla stringa, `%d` dla liczby całkowitej, `%f` dla liczby zmiennoprzecinkowej. Następnie podajemy po znaku % krotkę z wartościami do wstawienia.

- kolejnościowe
- nazwane
- typy: `string`, `int`, `float`
- operatory na stringu

```
imie = 'Hori Kamatani'
wiek = 18

def get_imie(imie):
    return imie

print('My name %s!' % imie)

print("%s is %s years old" % (imie, wiek))
print('%s is %s years old' % (wiek, imie))
print('%s is %10.1f years old' % (imie, wiek))
print('%s is %.1f years old' % (imie, wiek))
print('%s is %d years old' % (get_imie(imie), wiek))

print('%(imie)s is %(wiek)d years old' % {
    'wiek': wiek,
    'imie': imie,
})

print('My name %(imie)s.' % locals())
```

Metoda `.format()`

Wbudowana metoda `format` upraszcza nieco powyższy schemat. Zamiast operatora z procentem, używamy w tekście stringu `{}`, następnie na tym stringu wywołujemy funkcję `format`, której argumentami są wartości do wstawienia do tekstu.

- `string`
- `int`
- `float`
- operatory na stringu
- jako parametry do `print("string", **args)`

```
imie = 'Hori Kamatani'
wiek = 18

print('{imie} ma {wiek} lat'.format(
        imie=imie,
        wiek=wiek))

print('{wiek} ma {imie} lat'.format(**locals()))

print('Hej mam na imie {} i mam {} lat'.format(imie, wiek))

>>> print('Hej mam na imie {0} i mam {1} lat'.format(imie, wiek))
Hej mam na imie Hori i mam 10 lat

>>> print('Hej mam na imie {1} i mam {0} lat'.format(imie, wiek))
Hej mam na imie 10 i mam Hori lat

>>> print('Hej mam na imie {1:.3} i mam {0:.3} lat'.format(float(wiek), imie))
```

```
Hej mam na imie Hor i mam 10.0 lat

>>> print('Hej mam na imie {1:.3} i mam {0:10.3} lat'.format(float(wiek), imie))
Hej mam na imie Hor i mam       10.0 lat
```

f-strings - Python >= 3.6

f-strings to rozwinięcie funkcji `format`. Jedyne co trzeba zrobić żeby umieścić zmienną w tekście to dodać przed stringiem `f` i w nawiasach klamrowych wpisać nazwę zmiennej (np. `f'to jest zmienna: {zmienna}'`).

- `f'{variable}'`
- `f'{self.field}'`
- `f'{datetime:%Y-%m-%d %H:%M}'`

```
import datetime

imie = 'Hori'
wiek = 18


def get_imie(imie):
    return imie

print(f'My name {imie}')
print(f'My name {get_imie()}, masz: {wiek} lat')


print(f'dzis jest: {datetime.datetime.now():%Y-%m-%d %H:%M}')

now = datetime.datetime.now
print(f'dzis jest: {now():%Y-%m-%d %H:%M}')
```

Przykład z życia

Kod podatny jest na SQL Injection. W praktyce skorzystaj z funkcji `prepare`.

```
sql_query = f"""

    SELECT id, username, email
    FROM users
    WHERE 'username' = '{username}'
    AND 'password' = '{password}'

"""
```

Więcej informacji

- https://pyformat.info - Formatowanie stringów w Python

`pprint`

```
from pprint import pprint

data = [{'first_name': 'José', 'last_name': 'Jiménez'}, {'first_name': 'Max',
'last_name': 'Peck'}, {'first_name': 'Ivan', 'last_name': 'Ivanovic'}]

pprint(data)
```

Zadania kontrolne

Powielanie napisów

Napisz program, który wczyta od użytkownika pewien napis, a następnie wyświetli 5 kopii tego napisu, każda w osobnej linii. Napisz doctest do takiej funkcji. Napisz trzy wersje tego programu:

- wykorzystując `range()`
- wykorzystując pętlę `while`
- wykorzystując właściwości mnożenia stringów `print('ciag znakow' * 5)`

Przeliczanie temperatury

Woda zamarza przy 32 stopniach Fahrenheita, a wrze przy 212 stopniach Fahrenheita. Napisz program, który wyświetli tabelę przeliczeń stopni Celsjusza na stopnie Fahrenheita w zakresie od –20 do +40 stopni Celsjusza (co 5 stopni). Pamiętaj o wyświetlaniu znaku plus/minus przy temperaturze. Oczywiście napisz testy do rozwiązania.

- Zrób aby znak plus lub minus był zawsze wyświetlany.
- Zrób aby tabelka była stałej szerokości.

Podpowiedź

: - Fahrenheit to Celsius: (°F - 32) / 1.8 = °C - Celsius to Fahrenheit: (°C * 1.8) + 32 = °F - skorzystaj z funkcji `range()`

https://support.smartbear.com/testcomplete/docs/scripting/working-with/strings/python.html