Katando Python II

isra@miscorreos.org
@kamaxeon

Agenda

- **args y kwargs**
- **comprehension**
- **yield**

args

```
def example_args(*args):
    for num, value in enumerate(args, 1):
        print('Argument number {}, with value {}'.format(num, value))

example_args('foo', 'bar', 2, None, True)

Argument number 1, with value foo
Argument number 2, with value bar
Argument number 3, with value 2
Argument number 4, with value None
Argument number 5, with value True
```

kwargs

```
def example_kwargs(**kwargs):
    for num, (key, value) in enumerate(kwargs.items(), 1):
        print('Kwarg {}, key: {}, value {}'.format(num, key, value))

example_kwargs(name='Guido', surname='van Rossum')
```

```
Kwarg 1, key: surname, value van Rossum
Kwarg 2, key: name, value Guido
```

args y kwargs

```
def example(*args, **kwargs):
    print('Arguments list')
    for num, value in enumerate(args, 1):
        print('Arg {}, value {}'.format(num, value))
    print('Keyword Arguments list')
    for num, (key, value) in enumerate(kwargs.items(), 1):
        print('Kwarg {}, key: {}, value {}'.format(num, key, value))


example('python', name='Guido', surname='van Rossum', 'rule')
SyntaxError: non-keyword arg after keyword arg
```

args y kwargs

```
def example(*args, **kwargs):
    print('Arguments list')
    for num, value in enumerate(args, 1):
        print('Arg {}, value {}'.format(num, value))
    print('Keyword Arguments list')
    for num, (key, value) in enumerate(kwargs.items(), 1):
        print('Kwarg {}, key: {}, value {}'.format(num, key, value))

example('python', 'rules', name='Guido', surname='van Rossum')
Arguments list
Arg 1, value python
Arg 2, value rules
Keyword Arguments list
Kwarg 1, key: surname, value van Rossum
Kwarg 2, key: name, value Guido
```

List comprehension

```
# Calculate the square of every element of a list
origin = [1, 3, 4, 6, 8]
square = []
for element in origin:
    square.append(element**2)

square2 = [element**2 for element in origin]

print(square)
print(square2)

[1, 9, 16, 36, 64]
[1, 9, 16, 36, 64]
```

List comprehension using conditional

```
# Get a list of even from the list origin
origin = [1, 2, 3, 4]
even = []
for element in origin:
    if element % 2 == 0:
        even.append(element)

even2 = [element for element in origin if element % 2 == 0]

print(even)
print(even2)
[2, 4]
[2, 4]
```

List comprehension using multiples conditionals

```
# Get a list of even and it divided by 6 from 0 to 49
divided = []

for x in range(50):
    if x%2 == 0 and x%6 == 0 :
            divided.append(x)
divided2 = [x for x in range(50) if x % 2 == 0 if x % 6 == 0]

print(divided)
print(divided2)
[0, 6, 12, 18, 24, 30, 36, 42, 48]
[0, 6, 12, 18, 24, 30, 36, 42, 48]
```

Nested list comprehension

```
# Flat matrix
matrix = [[1, 2, 3],
          [4, 5, 6],
          [7, 8, 9]]

flatten_matrix = []
for row in range(len(matrix)):
    for col in range(len(matrix[row])):
        flatten_matrix.append(matrix[row][col])

flatten_matrix2 = [y for x in matrix for y in x]

print(flatten_matrix)
print(flatten_matrix2)

[1, 2, 3, 4, 5, 6, 7, 8, 9]
[1, 2, 3, 4, 5, 6, 7, 8, 9]
```

Types of comprehension

```
>>> type( [x for x in [1,2,3]] )
<class 'list'>
>>> type( {x for x in [1,2,3]} )
<class 'set'>
>>> type( (x for x in [1,2,3]) )
<class 'generator'>
>>> type( {x:'foo' for x in [1,2,3]} )
<class 'dict'>
```

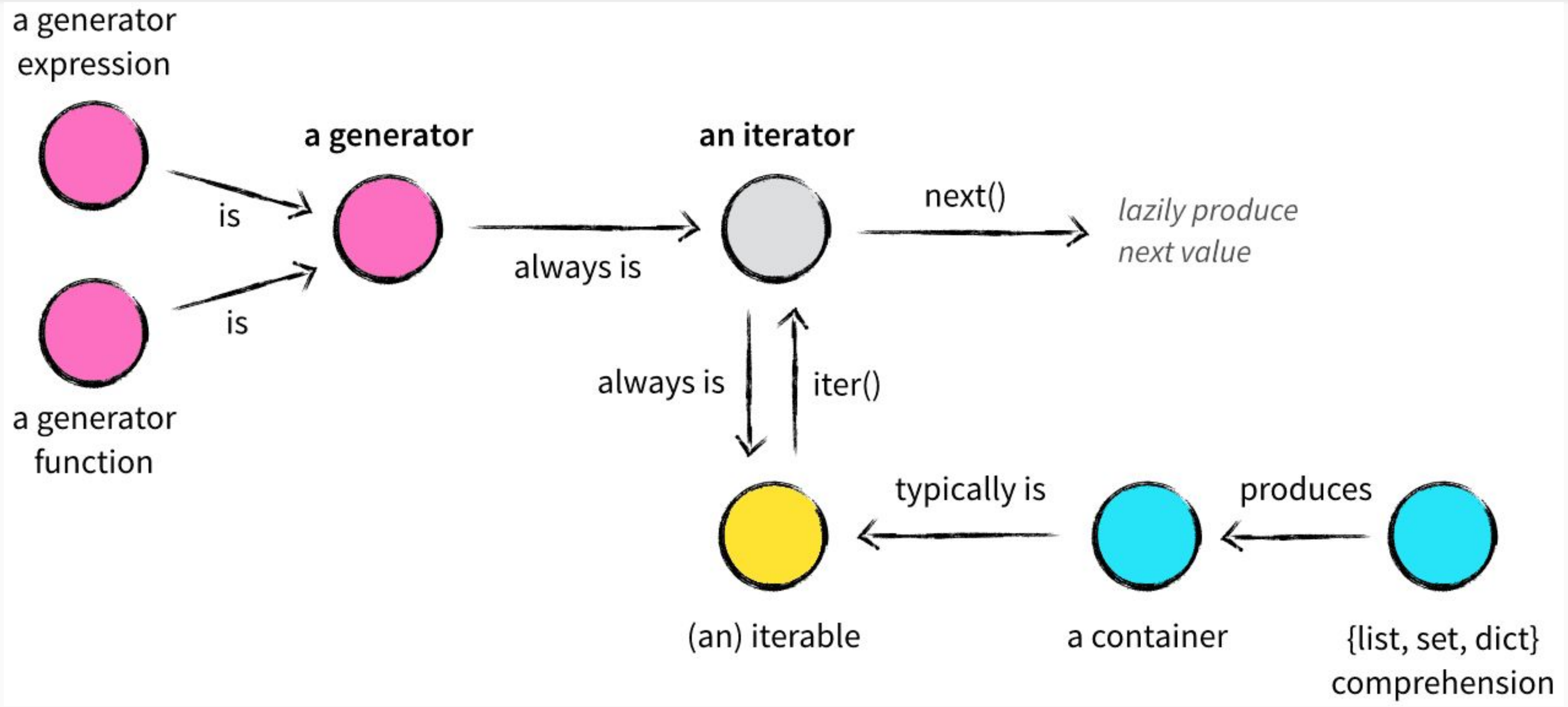
My common use of comprehension

```
>>> sum(x**2 for x in [1,2,3])
14
>>> min(x for x in [1,2,3,4] if x%2 == 0)
2
>>> any(x%2 == 0 for x in [1,2,3,4])
True
>>> all(x%2 == 0 for x in [1,2,3,4])
False
```

yield

```
>>> def counterGenerator():
...     i = 0
...     while True:
...         yield i
...         i += 1
...
>>> my_counter = counterGenerator()
>>> next(my_counter)
0
>>> next(my_counter)
1
>>> next(my_counter)
2
>>> next(my_counter)
3
```

yield



https://nvie.com/posts/iterators-vs-generators/

Questions ?

References

- https://www.datacamp.com/community/tutorials/python-list-comprehension
- https://snakify.org/en/lessons/two_dimensional_lists_arrays/
- https://www.digitalocean.com/community/tutorials/how-to-use-args-and-kwargs-in-python-3
- https://python-3-patterns-idioms-test.readthedocs.io/en/latest/Comprehensions.html