Функциональные возможности языка Python.

Выполнил: Ордиянц Эрик
Группа: ИУ5-36Б
Дата: 18.12.24г.

Описание задания:
lab_python_fp · ugapanyuk/BKIT_2022 Wiki · GitHub

Код программы:

1 Задание:

```python
def field(items, *args):
    assert len(args) > 0
    if len(args) == 1:
        key = args[0]
        for item in items:
            value = item.get(key)
            if value is not None:
                yield value
    else:
        for item in items:
            result = {key: item.get(key) for key in args if item.get(key) is not None}
            if result:
                yield result

if __name__ == '__main__':
    goods = [
        {'title': 'Ковер', 'price': 2000, 'color': 'green'},
        {'title': 'Диван для отдыха', 'color': 'black'}
    ]
    print(list(field(goods, 'title')))
    print(list(field(goods, 'title', 'price', 'color')))
```

2 Задание:

```python
import random

def gen_random(num_count, begin, end):
    for _ in range(num_count):
        yield random.randint(begin, end)

if __name__ == '__main__':
    print(list(gen_random(5, 1, 3)))
```

3 Задание:

```
class Unique(object):
    def __init__(self, items, **kwargs):
        self.items = iter(items)
        self.seen = set()
        self.ignore_case = kwargs.get('ignore_case', False)

    def __iter__(self):
        return self

    def __next__(self):
        while True:
            item = next(self.items)
            compare_item = item.lower() if self.ignore_case and isinstance(item, str)
else item
            if compare_item not in self.seen:
                self.seen.add(compare_item)
                return item


if __name__ == '__main__':
    data = [1, 1, 1, 1, 1, 2, 2, 2, 2, 2]
    print(list(Unique(data)))

    from gen_random import gen_random
    data = gen_random(10, 1, 3)
    print(list(Unique(data)))

    data = ['a', 'A', 'b', 'B', 'a', 'A', 'b', 'B']
    print(list(Unique(data)))
    print(list(Unique(data, ignore_case=True)))
```

4 Задание:

```python
data = [4, -30, 100, -100, 123, 1, 0, -1, -4]

if __name__ == '__main__':
    result = sorted(data, reverse=True)
    print(result)

    result_with_lambda = sorted(data, key=lambda x: abs(x), reverse=True)
    print(result_with_lambda)

    result = sorted(data, reverse=False)
    print(result)

    result_with_lambda = sorted(data, key=lambda x: abs(x), reverse=False)
    print(result_with_lambda)
```

5 Задание:

```
from field import field
from gen_random import gen_random
from unique import Unique
from sort import data

def print_result(func):
    def wrapper(*args, **kwargs):
        result = func(*args, **kwargs)
        print(func.__name__)
        if isinstance(result, list):
            for item in result:
                if isinstance(item, dict):
                    for key, value in item.items():
                        print(f"{key} = {value}")
                else:
                    print(item)
        elif isinstance(result, dict):
            for key, value in result.items():
                print(f"{key} = {value}")
        else:
            print(result)
        return result
    return wrapper

@print_result
def test_field():
    goods = [
        {'title': 'Ковер', 'price': 2000, 'color': 'green'},
        {'title': 'Диван для отдыха', 'price': 5300, 'color': 'black'}
    ]
    return list(field(goods, 'title', 'price'))

@print_result
def test_gen_random():
    return list(gen_random(5, 1, 3))

@print_result
def test_unique():
    data = [1, 1, 2, 2, 3, 3, 4, 4]
    return list(Unique(data))

@print_result
def test_sort():
    result_no_lambda = sorted(data, key=abs, reverse=True)
    return result_no_lambda

@print_result
def test_sort_with_lambda():
    result_with_lambda = sorted(data, key=lambda x: abs(x), reverse=True)
    return result_with_lambda

if __name__ == "__main__":
```

```
    test_field()
    test_gen_random()
    test_unique()
    test_sort()
    test_sort_with_lambda()
```

6 Задание:

```python
import time
from contextlib import contextmanager

class cm_timer_1:
    def __enter__(self):
        self.start_time = time.time()
        return self

    def __exit__(self, exc_type, exc_val, exc_tb):
        end_time = time.time()
        print(f"time: {end_time - self.start_time:.2f}")

@contextmanager
def cm_timer_2():
    start_time = time.time()
    try:
        yield
    finally:
        end_time = time.time()
        print(f"time: {end_time - start_time:.2f}")


if __name__ == '__main__':
    with cm_timer_1():
        time.sleep(5.5)

    with cm_timer_2():
        time.sleep(3.2)
```

7 Задание:

```python
import json
import sys
import random
from contextlib import contextmanager
import time


def print_result(func):
    def wrapper(*args, **kwargs):
        result = func(*args, **kwargs)
        print(func.__name__)
        if isinstance(result, list):
            for item in result:
                print(item)
        elif isinstance(result, dict):
            for key, value in result.items():
                print(f"{key} = {value}")
        else:
            print(result)
        return result
    return wrapper
```

```python
class cm_timer_1:
    def __enter__(self):
        self.start_time = time.time()
        return self

    def __exit__(self, exc_type, exc_val, exc_tb):
        end_time = time.time()
        print(f"time: {end_time - self.start_time:.2f} seconds")


path = sys.argv[1] if len(sys.argv) > 1 else "data_light.json"
with open(path, encoding="utf-8") as f:
    data = json.load(f)

@print_result
def f1(arg):
    return sorted(set(job['job-name'].strip().lower().capitalize() for job in arg))

@print_result
def f2(arg):
    return list(filter(lambda x: x.lower().startswith('программист'), arg))

@print_result
def f3(arg):
    return list(map(lambda x: f"{x} с опытом Python", arg))

@print_result
def f4(arg):
    salaries = [random.randint(100000, 200000) for _ in arg]
    return [f"{job}, зарплата {salary} руб." for job, salary in zip(arg, salaries)]

if __name__ == '__main__':
    with cm_timer_1():
        f4(f3(f2(f1(data))))
```

Снимки экрана:
1 Задание:

```
C:\Users\Urbech\Desktop\labs\lab 3-4>python 1ex.py
['Ковер', 'Диван для отдыха']
['green', 'black']
[{'title': 'Ковер', 'price': 2000}, {'title': 'Диван для отдыха'}]
```

2 Задание:

```
C:\Users\Urbech\Desktop\labs\lab 3-4>python 2ex.py
[87, 34, 15, 61]
```

3 Задание:

```
C:\Users\Urbech\Desktop\labs\lab 3-4>python 3ex.py
1
2
3
a
A
b
B
a
b
```

4 Задание:

```
C:\Users\Urbech\Desktop\labs\lab 3-4>python 4ex.py
[123, 100, -100, -30, 4, -4, 1, -1, 0]
[0, 1, -1, 4, -4, -30, 100, -100, 123]
```

5 Задание:

```
C:\Users\Urbech\Desktop\labs\lab 3-4>python 5ex.py
!!!!!!!!
test_1
1
test_2
iu5
test_3
a = 1
b = 2
test_4
1
2
```

6 Задание:

```
C:\Users\Urbech\Desktop\labs\lab 3-4>python 6ex.py
Using cm_timer_1:
time: 2.00
Using cm_timer_2:
time: 2.00
```

7 Задание:

```
C:\Users\Urbech\Desktop\labs\lab 3-4>python 7ex.py
f1
1c программист
2-ой механик
3-ий механик
4-ый механик
4-ый электромеханик
Asic специалист
Javascript разработчик
Rtl специалист
Web-программист
Web-разработчик
[химик-эксперт
Автожестянщик
Автоинструктор
Автомаляр
Автомойщик
Автор студенческих работ по различным дисциплинам
Автослесарь
Автослесарь - моторист
Юрист (специалист по сопровождению международных договоров, английский - раз
Юрист волонтер
Юристконсульт
f2
Программист
Программист / senior developer
Программист 1с
Программист c#
Программист c++
Программист c++/c#/java
Программист/ junior developer
Программист/ технический специалист
Программистр-разработчик  информационных систем
f3
Программист с опытом Python
Программист / senior developer с опытом Python
Программист 1с с опытом Python
Программист c# с опытом Python
Программист c++ с опытом Python
Программист c++/c#/java с опытом Python
Программист/ junior developer с опытом Python
Программист/ технический специалист с опытом Python
Программистр-разработчик  информационных систем с опытом Python
```

```
f4
Программист с опытом Python, зарплата 143666 руб.
Программист / senior developer с опытом Python, зарплата 151981 руб.
Программист 1с с опытом Python, зарплата 164241 руб.
Программист c# с опытом Python, зарплата 174835 руб.
Программист c++ с опытом Python, зарплата 175074 руб.
Программист c++/c#/java с опытом Python, зарплата 119420 руб.
Программист/ junior developer с опытом Python, зарплата 153200 руб.
Программист/ технический специалист с опытом Python, зарплата 191643 руб.
Программистр-разработчик  информационных систем с опытом Python, зарплата 18
time: 0.09 seconds
```