РК2

Студент: Гайнуллин А.М.

Группа: ИУ5-32Б

Текст программы

Файл main.py

from operator import itemgetter

from collections import Counter

from typing import Dict, List

import data

from classes import Computer, ComputerDisplay, Display

# Соединение данных один-ко-многим

def getOneToMany(displays: List[ComputerDisplay],computers: List[Computer]):

return [(computer.name, computer.cost, display.name)

for display in displays

for computer in computers

if computer.display\_id == display.id]

# Соединение данных многие-ко-многим

def getManyToMany(displays: List[Display], computerDisplays: List[ComputerDisplay], computers: List[Computer]):

manyToManyTemp = [(display.name, computerDisplay.display\_id, computerDisplay.computer\_id)

for display in displays

for computerDisplay in computerDisplays

if display.id == computerDisplay.display\_id]

return [(computer.id, computer.name, computer.cost, display\_name)

for display\_name, display\_id, computer\_id in manyToManyTemp

for computer in computers if computer.id == computer\_id]

def getDisplayForEachComputer(manyToMany, computers: List[Computer]):

result: Dict[str, List[str]] = {}

for computer in computers:

displaysWithComputer = [m[3] for m in filter(lambda i: i[0] == computer.id, manyToMany)]

result[computer.name] = displaysWithComputer

return result

def getQuantityOfComputersInDisplays(displays: List[ComputerDisplay], computers: List[Computer]):

displaysCounter = Counter([computer.display\_id for computer in computers])

return [(l.name, displaysCounter[l.id]) for l in computers]

def main():

# «Дисплейный класс» и «Компьютер» связаны соотношением один-ко-многим.

# Выведите список всех связанных компьютеров и дисплеев, отсортированный по названию компьютера,

# сортировка по дисплеям произвольная.

oneToMany = getOneToMany(data.displays, data.computers)

# Соединение данных многие-ко-многим

manyToMany = getManyToMany(data.displays, data.computerDisplays, data.computers)

print('Задание Б1')

resultA = sorted(oneToMany, key=itemgetter(0))

print(resultA)

# «Дисплей» и «Компьютер» связаны соотношением один-ко-многим.

# Выведите список дисплеев с количеством комьютеров в каждом дисплее, отсортированный по количеству компьютеров.

print('\nЗадание Б2')

result\_b\_unsorted = getQuantityOfComputersInDisplays(data.displays, data.computers)

print(sorted(result\_b\_unsorted, key=itemgetter(1), reverse=True))

# «Дисплей» и «Компьютер связаны соотношением многие-ко-многим.

# Выведите список всех компьютеров, у которых название заканчивается на «r», и названия их дисплеев.

print('\nЗадание Б3')

print(getDisplayForEachComputer(manyToMany, list(filter(lambda x: x.name.endswith('r'), data.computers))))

if \_\_name\_\_ == '\_\_main\_\_':

main()

Файл classes.py

class Computer:

""" Класс компьютер"""

def \_\_init\_\_(self, id\_: int, name: str, cost: int, display\_id: int):

self.id = id\_

self.name = name

self.cost = cost

self.display\_id = display\_id

class Display:

"""Дисплейный класс"""

def \_\_init\_\_(self, id\_: int, name: str):

self.id = id\_

self.name = name

class ComputerDisplay:

"""

'Компьютеры дисплейного класса' для реализации

связи многие-ко-многим

"""

def \_\_init\_\_(self, computer\_id: int, display\_id: int):

self.computer\_id = computer\_id

self.display\_id = display\_id

Файл data.py

from classes import Computer, ComputerDisplay, Display

displays = [

Display(1, 'Первый дисплей'),

Display(2, 'Второй дисплей'),

Display(3, 'Третий дисплей'),

]

computers = [

Computer(1, 'Admin', 100000, 1),

Computer(2, 'Hero', 60000, 1),

Computer(3, 'CoolComputer', 75000, 2),

Computer(4, 'MasterOfComputers', 80000, 2),

Computer(5, 'PusherForever',167000, 2),

]

computerDisplays = [

ComputerDisplay(1, 1),

ComputerDisplay(1, 3),

ComputerDisplay(2, 1),

ComputerDisplay(2, 3),

ComputerDisplay(3, 2),

ComputerDisplay(4, 2),

ComputerDisplay(4, 3),

ComputerDisplay(5, 2),

]

Файл tests.py

import unittest

import data

from main import getOneToMany, getManyToMany, getQuantityOfComputersInDisplays, getDisplayForEachComputer

class TestFunctions(unittest.TestCase):

def setUp(self):

self.data = data

def testGetOneToMany(self):

want = [('Admin', 100000, 'Первый дисплей'), ('Hero', 60000, 'Первый дисплей'),

('CoolComputer', 75000, 'Второй дисплей'), ('MasterOfComputers', 80000, 'Второй дисплей'),

('PusherForever', 167000, 'Второй дисплей')]

actual = getOneToMany(self.data.displays, self.data.computers)

self.assertCountEqual(want, actual)

def testGetManyToMany(self):

want = [(1, 'Admin', 100000, 'Первый дисплей'), (2, 'Hero', 60000, 'Первый дисплей'),

(3, 'CoolComputer', 75000, 'Второй дисплей'), (4, 'MasterOfComputers', 80000, 'Второй дисплей'),

(5, 'PusherForever', 167000, 'Второй дисплей'), (1, 'Admin', 100000, 'Третий дисплей'),

(2, 'Hero', 60000, 'Третий дисплей'), (4, 'MasterOfComputers', 80000, 'Третий дисплей')]

actual = getManyToMany(self.data.displays, self.data.computerDisplays, self.data.computers)

self.assertCountEqual(want, actual)

def testGetQuantityOfComputersInDisplays(self):

want = [('Admin', 2), ('Hero', 3), ('CoolComputer', 0), ('MasterOfComputers', 0), ('PusherForever', 0)]

actual = getQuantityOfComputersInDisplays(self.data.displays, self.data.computers)

self.assertCountEqual(want, actual)

def testGetDisplayForEachComputer(self):

manyToMany = getManyToMany(self.data.displays, self.data.computerDisplays, self.data.computers)

want = {'Admin': ['Первый дисплей', 'Третий дисплей'], 'Hero': ['Первый дисплей', 'Третий дисплей'],

'CoolComputer': ['Второй дисплей'],

'MasterOfComputers': ['Второй дисплей', 'Третий дисплей'], 'PusherForever': ['Второй дисплей']}

actual = getDisplayForEachComputer(manyToMany, self.data.computers)

self.assertDictEqual(want, actual)

Результаты выполнения

Задание Б1

[('Admin', 100000, 'Первый дисплей'), ('CoolComputer', 75000, 'Второй дисплей'), ('Hero', 60000, 'Первый дисплей'), ('MasterOfComputers', 80000, 'Второй дисплей'), ('PusherForever', 167000, 'Второй дисплей')]

Задание Б2

[('Hero', 3), ('Admin', 2), ('CoolComputer', 0), ('MasterOfComputers', 0), ('PusherForever', 0)]

Задание Б3

{'CoolComputer': ['Второй дисплей'], 'PusherForever': ['Второй дисплей']}

Результаты выполнения тестов

....

----------------------------------------------------------------------

Ran 4 tests in 0.000s

OK