Микулин Михаил

ИУ5-36Б

Рубежный контроль №2

Текст программы

tasks.py

def first\_task(conductors, orchestras):

r = [(

conductor.fio, orchestra.name)

for conductor in conductors

for orchestra in orchestras

if (conductor.orchestra\_id == orchestra.id)

and (conductor.fio[-2:] == "ни")]

return r

def second\_task(conductors, orchestras):

r = [[\_.name, 0, 0] for \_ in orchestras]

for conductor in conductors:

r[conductor.orchestra\_id - 1][1] += conductor.years\_of\_experience

r[conductor.orchestra\_id - 1][2] += 1

r = [(\_[0], \_[1]/\_[2]) for \_ in r]

r.sort(key=lambda r: r[1])

return r

def third\_task(conductors, orchestras, many\_to\_many):

r = [(\_.name, []) for \_ in orchestras]

connections = [(orchestras[connection.orchestra\_id-1].name, conductors[connection.conductor\_id-1].fio) for connection in many\_to\_many]

for \_ in connections:

for elm in r:

if elm[0] == \_[0]:

elm[1].append(\_[1])

to\_be\_del = []

for i in range (len(r)):

if r[i][0][0] != "R":

to\_be\_del.append(i)

to\_be\_del.reverse()

for i in to\_be\_del:

del r[i]

return r

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

main()

tests.py

import unittest

from src.conductor import \*

from src.orchestra import \*

from src.connection import \*

import tasks

class TestRK(unittest.TestCase):

orchestras = [

Orchestra(1, "Royal Concertgebouw Orchestra", "Amsterdam"),

Orchestra(2, "Berlin Philharmonic Orchestra", "Berlin"),

Orchestra(3, "Los Angeles Philharmonic", "Los-Angeles")

]

conductors = [

Conductor(1, "Томас Бичем", 1, 10),

Conductor(2, "Карло Мария Джулини", 2, 15),

Conductor(3, "Артуро Тосканини", 3, 16),

Conductor(4, "Вильгельм Фуртвенглер", 2, 19),

Conductor(5, "Николаус Арнонкур", 1, 12)

]

many\_to\_many = [

ConductorOrchestra(3, 1),

ConductorOrchestra(2, 1),

ConductorOrchestra(1, 2),

ConductorOrchestra(1, 3),

ConductorOrchestra(4, 2)

]

def test\_1(self):

self.assertEqual(tasks.first\_task(self.conductors, self.orchestras), [('Карло Мария Джулини', 'Berlin Philharmonic Orchestra'), ('Артуро Тосканини', 'Los Angeles Philharmonic')], 'wrong')

def test\_2(self):

self.assertEqual(tasks.second\_task(self.conductors, self.orchestras), [('Royal Concertgebouw Orchestra', 11.0), ('Los Angeles Philharmonic', 16.0), ('Berlin Philharmonic Orchestra', 17.0)], 'wrong')

def test\_3(self):

self.assertEqual(tasks.third\_task(self.conductors, self.orchestras, self.many\_to\_many), [('Royal Concertgebouw Orchestra', ['Артуро Тосканини', 'Карло Мария Джулини'])], 'wrong')

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

unittest.main()

Результат программы

