**Московский государственный технический**

**университет им. Н.Э. Баумана.**

Факультет «Информатика и управление»

Кафедра ИУ5. Курс «Базовые компоненты интернет технологий»

Отчет по рубежному контролю №2

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-31Б |  | Гапанюк Ю.Е. |
| Шагиахметов А.Л. |  |  |
| Подпись и дата: 19.12.2021 |  | Подпись и дата: |

г. Москва, 2021 г.

Текст программы:

**from** operator **import** itemgetter  
  
  
**class** CD:  
 *"""CD-диск"""* **def** \_\_init\_\_(self, id, names, cost, dep\_id):  
 self.id = id  
 self.names = names  
 self.cost = cost  
 self.dep\_id = dep\_id  
  
  
**class** Lib:  
 *"""Библиотека CD-дисков"""* **def** \_\_init\_\_(self, id, name):  
 self.id = id  
 self.name = name  
  
  
**class** CDLib:  
 *"""  
 'Сотрудники отдела' для реализации  
 связи многие-ко-многим  
 """* **def** \_\_init\_\_(self, dep\_id, emp\_id):  
 self.dep\_id = dep\_id  
 self.emp\_id = emp\_id  
  
  
*# Библиотека CD-дисков*bibl = [  
 Lib(1, **'vip- библиотека Ласточка'**),  
 Lib(2, **'библиотека Тутик'**),  
 Lib(3, **'vip- библиотека Подслушка'**),  
 Lib(11, **'библиотека Читай'**),  
 Lib(22, **'библиотека Умник'**),  
 Lib(33, **'vip- библиотека CD'**),  
]  
  
*# CD-диски*cds = [  
 CD(1, **'диск Глаша'**, 2500, 1),  
 CD(2, **'диск Лямур'**, 3500, 2),  
 CD(3, **'диск Киберпанк'**, 400, 3),  
 CD(4, **'диск Стас Михайлов'**, 3000, 3),  
 CD(5, **'диск Брежнева'**, 250, 3),  
]  
  
cds\_bibl = [  
 CDLib(1, 1),  
 CDLib(2, 2),  
 CDLib(3, 3),  
 CDLib(3, 4),  
 CDLib(3, 5),  
 CDLib(11, 1),  
 CDLib(22, 2),  
 CDLib(33, 3),  
 CDLib(33, 4),  
 CDLib(33, 5),  
]  
  
  
**def** sorting\_by\_name(table):  
 **return** sorted(table, key=itemgetter(2))  
  
  
**def** sorting\_by\_sum\_size(table, bibl):  
 res\_12\_unsorted = []  
 *# Перебираем все каталоги* **for** d **in** bibl:  
 *# Список файлов каталога* d\_cds = list(filter(**lambda** i: i[2] == d.name, table))  
 *# Если библиотека не пустая* **if** len(d\_cds) > 0:  
 *# Стоимость дисков* d\_costs = [cost **for** \_, cost, \_ **in** d\_cds]  
 *# Суммарная стоимость дисков* d\_costs\_sum = sum(d\_costs)  
 res\_12\_unsorted.append((d.name, d\_costs\_sum))  
  
 *# Сортировка по суммарной стоимости дисков* **return** sorted(res\_12\_unsorted, key=itemgetter(1), reverse=**True**)  
  
  
**def** output\_files\_of\_catalogs\_with\_PAPKA2(table, bibl):  
 res\_13 = {}  
 *# Перебираем все библиотеки* **for** d **in** bibl:  
 **if 'vip-' in** d.name:  
 *# Список дисков в библиотеке* d\_cds = list(filter(**lambda** i: i[2] == d.name, table))  
 d\_cds\_names = [x **for** x, \_, \_ **in** d\_cds]  
 res\_13[d.name] = d\_cds\_names  
  
 **return** res\_13  
  
  
**def** main():  
 *"""Основная функция"""  
  
 # Соединение данных один-ко-многим* one\_to\_many = [(e.names, e.cost, d.name)  
 **for** d **in** bibl  
 **for** e **in** cds  
 **if** e.dep\_id == d.id]  
  
 *# Соединение данных многие-ко-многим* many\_to\_many\_temp = [(d.name, ed.dep\_id, ed.emp\_id)  
 **for** d **in** bibl  
 **for** ed **in** cds\_bibl  
 **if** d.id == ed.dep\_id]  
  
 many\_to\_many = [(e.names, e.cost, dep\_name)  
 **for** dep\_name, dep\_id, emp\_id **in** many\_to\_many\_temp  
 **for** e **in** cds **if** e.id == emp\_id]  
  
 print(**'Задание А1'**)  
 print(sorting\_by\_name(one\_to\_many))  
  
 print(**'\nЗадание А2'**)  
 print(sorting\_by\_sum\_size(one\_to\_many, bibl))  
  
 print(**'\nЗадание А3'**)  
 print(output\_files\_of\_catalogs\_with\_PAPKA2(many\_to\_many, bibl))  
  
  
**if** \_\_name\_\_ == **'\_\_main\_\_'**:  
 main()

Результат выполнения программы:

**Задание А1**

[('диск Глаша', 2500, 'vip- библиотека Ласточка'), ('диск Киберпанк', 400, 'vip- библиотека Подслушка'), ('диск Стас Михайлов', 3000, 'vip- библиотека Подслушка'), ('диск Брежнева', 250, 'vip- библиотека Подслушка'), ('диск Лямур', 3500, 'библиотека Тутик')]

**Задание А2**

[('vip- библиотека Подслушка', 3650), ('библиотека Тутик', 3500), ('vip- библиотека Ласточка', 2500)]

**Задание А3**

{'vip- библиотека Ласточка': ['диск Глаша'], 'vip- библиотека Подслушка': ['диск Киберпанк', 'диск Стас Михайлов', 'диск Брежнева'], 'vip- библиотека CD': ['диск Киберпанк', 'диск Стас Михайлов', 'диск Брежнева']}

Текст программы тестирования:

**from** main **import** CD, Lib, CDLib, sorting\_by\_name, sorting\_by\_sum\_size, output\_files\_of\_catalogs\_with\_PAPKA2  
**import** unittest  
  
  
**class** Tests(unittest.TestCase):  
 **def** setUp(self):  
  
 self.bibl = [  
 Lib(1, **'vip- библиотека Ласточка'**),  
 Lib(2, **'библиотека Тутик'**),  
 Lib(3, **'vip- библиотека Подслушка'**),  
 Lib(11, **'библиотека Читай'**),  
 Lib(22, **'библиотека Умник'**),  
 Lib(33, **'vip- библиотека CD'**),  
 ]  
 *# Файлы* self.cds = [  
 CD(1, **'диск Глаша'**, 2500, 1),  
 CD(2, **'диск Лямур'**, 3500, 2),  
 CD(3, **'диск Киберпанк'**, 400, 3),  
 CD(4, **'диск Стас Михайлов'**, 3000, 3),  
 CD(5, **'диск Брежнева'**, 250, 3),  
 ]  
 self.cds\_bibl = [  
 CDLib(1, 1),  
 CDLib(2, 2),  
 CDLib(3, 3),  
 CDLib(3, 4),  
 CDLib(3, 5),  
 CDLib(11, 1),  
 CDLib(22, 2),  
 CDLib(33, 3),  
 CDLib(33, 4),  
 CDLib(33, 5),  
 ]  
 *# Соединение данных один-ко-многим* self.one\_to\_many = [(e.names, e.cost, d.name)  
 **for** d **in** bibl  
 **for** e **in** cds  
 **if** e.dep\_id == d.id]  
  
 *# Соединение данных многие-ко-многим* self.many\_to\_many\_temp = [(d.name, ed.dep\_id, ed.emp\_id)  
 **for** d **in** bibl  
 **for** ed **in** cds\_bibl  
 **if** d.id == ed.dep\_id]  
  
 self.many\_to\_many = [(e.names, e.cost, dep\_name)  
 **for** dep\_name, dep\_id, emp\_id **in** many\_to\_many\_temp  
 **for** e **in** cds **if** e.id == emp\_id]  
  
 **def** test\_sorting\_by\_name(self):  
 result = sorting\_by\_name(self.one\_to\_many)  
 desired\_result = [(**'диск Глаша'**, 2500, **'vip- библиотека Ласточка'**),  
 (**'диск Киберпанк'**, 400, **'vip- библиотека Подслушка'**),  
 (**'диск Стас Михайлов'**, 3000, **'vip- библиотека Подслушка'**),  
 (**'диск Брежнева'**, 250, **'vip- библиотека Подслушка'**),  
 (**'диск Лямур'**, 3500, **'библиотека Тутик'**)]  
  
 self.assertEqual(result, desired\_result)  
  
 **def** test\_sorting\_by\_sum(self):  
 result = sorting\_by\_sum\_size(self.one\_to\_many, self.catalogs)  
 desired\_result = [(**'vip- библиотека Подслушка'**, 3650), (**'библиотека Тутик'**, 3500),  
 (**'vip- библиотека Ласточка'**, 2500)]  
 self.assertEqual(result, desired\_result)  
  
 **def** test\_output\_PAPKA2(self):  
 result = output\_files\_of\_catalogs\_with\_PAPKA2(self.many\_to\_many, self.catalogs)  
 desired\_result = {**'vip- библиотека Ласточка'**: [**'диск Глаша'**],  
 **'vip- библиотека Подслушка'**: [**'диск Киберпанк'**, **'диск Стас Михайлов'**, **'диск Брежнева'**],  
 **'vip- библиотека CD'**: [**'диск Киберпанк'**, **'диск Стас Михайлов'**, **'диск Брежнева'**]}  
 self.assertEqual(result, desired\_result)

Пример работы тестирования:

