|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

|  |  |
| --- | --- |
| ФАКУЛЬТЕТ | ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ |

|  |  |
| --- | --- |
| КАФЕДРА | СИСТЕМЫ ОБРАБОТКИ ИНФОРМАЦИИ И УПРАВЛЕНИЯ |

Отчет по рубежному контролю №1.

Вариант №14(А)

Студент Мартынова Полина Владимировна.

*фамилия, имя, отчество*

Группа ИУ5-51Б.

Студент 18.10.2021 **Мартынова П.В.**

*подпись, дата фамилия, и.о.*

Преподаватель 18.10.2021 **Гапанюк Ю.Е.**  *подпись, дата фамилия, и.о.*

2021г.

**Задание:**

1) "CD-диск" и "Библиотека CD-дисков" связаны соотношением один-ко-многим. Выведите список всех связанных дисков и библиотек, отсортированный по библиотекам, сортировка по дискам произвольная.

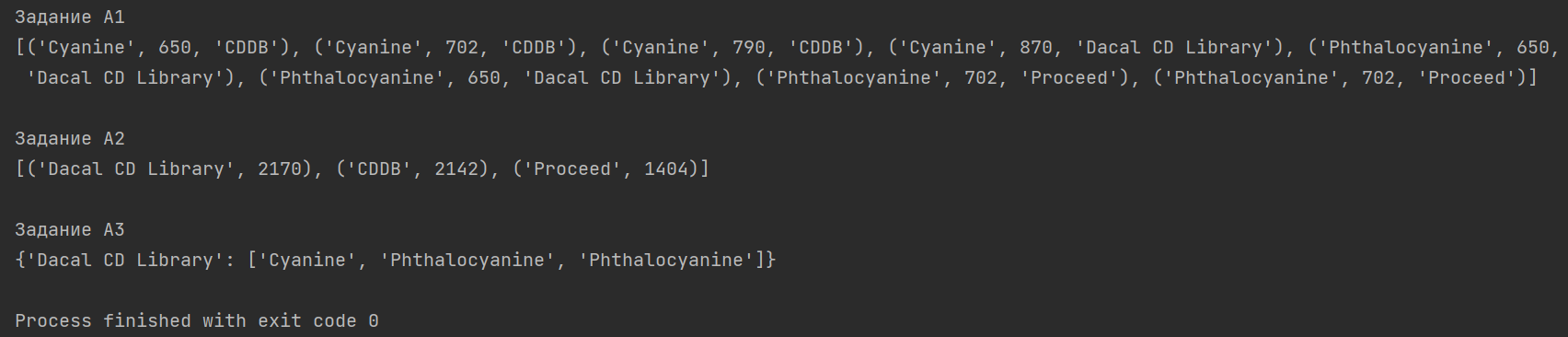
2) " CD-диск " и " Библиотека CD-дисков " связаны соотношением один-ко-многим. Выведите список дисков с суммарной ёмкостью в библиотеке дисков, отсортированный по суммарному занимаемому месту.

3) " CD-диск " и " Библиотека CD-дисков " связаны соотношением многие-ко-многим. Выведите список всех библиотек, у которых в названии присутствует слово «Library», и список их дисков.

Код:

# используется для сортировки  
from operator import itemgetter  
  
class CD:  
 *"""cd-диск"""* def \_\_init\_\_(self, id, capacity, recording\_layer\_type, lib\_id):  
 self.id = id  
 self.capacity = capacity  
 self.recording\_layer\_type = recording\_layer\_type  
 self.lib\_id = lib\_id  
  
class CD\_lib:  
 *"""библиотека cd-дисков"""* def \_\_init\_\_(self, id, name):  
 self.id = id  
 self.name = name  
  
class LibDisk:  
 def \_\_init\_\_(self, lib\_id, disk\_id):  
 self.lib\_id = lib\_id  
 self.disk\_id = disk\_id  
  
# библиотеки cd-дисков  
libs = [  
 CD\_lib(1, 'CDDB'),  
 CD\_lib(2, 'Dacal CD Library'),  
 CD\_lib(3, 'Proceed')  
]  
  
# диски  
disks = [  
 CD(1, 650, 'Cyanine', 1),  
 CD(2, 702, 'Cyanine', 1),  
 CD(3, 790, 'Cyanine', 1),  
 CD(4, 870, 'Cyanine', 2),  
 CD(5, 650, 'Phthalocyanine', 2),  
 CD(6, 650, 'Phthalocyanine', 2),  
 CD(7, 702, 'Phthalocyanine', 3),  
 CD(8, 702, 'Phthalocyanine', 3),  
 CD(9, 650, 'Phthalocyanine', 4),  
 CD(10, 650, 'Phthalocyanine', 4)  
]  
  
lib\_disks = [  
 LibDisk(1, 1),  
 LibDisk(1, 2),  
 LibDisk(1, 3),  
 LibDisk(2, 4),  
 LibDisk(2, 5),  
 LibDisk(2, 6),  
 LibDisk(3, 7),  
 LibDisk(3, 8),  
 LibDisk(4, 9),  
 LibDisk(4, 10),  
]  
  
def main():  
 # Соединение данных один-ко-многим  
 one\_to\_many = [(e.recording\_layer\_type, e.capacity, d.name)  
 for d in libs  
 for e in disks  
 if e.lib\_id == d.id]  
  
 # Соединение данных многие-ко-многим  
 many\_to\_many\_temp = [(d.name, ed.lib\_id, ed.disk\_id)  
 for d in libs  
 for ed in lib\_disks  
 if d.id == ed.lib\_id]  
  
 many\_to\_many = [(e.recording\_layer\_type, e.capacity, lib\_name)  
 for lib\_name, lib\_id, disk\_id in many\_to\_many\_temp  
 for e in disks if e.id == disk\_id]  
  
 print('Задание А1')  
 res\_11 = sorted(one\_to\_many, key=itemgetter(2))  
 print(res\_11)  
  
 print('\nЗадание А2')  
 res\_12\_unsorted = []  
 # Перебираем все библиотеки  
 for d in libs:  
 # Список дисков в библиотеке  
 d\_disks = list(filter(lambda i: i[2] == d.name, one\_to\_many))  
 # Если библиотека не пустая  
 if len(d\_disks) > 0:  
 # ёмкости дисков в библиотеке  
 d\_capacitys = [capacity for \_, capacity, \_ in d\_disks]  
 # суммарная ёмкость дисков в библиотеке  
 d\_capacitys\_sum = sum(d\_capacitys)  
 res\_12\_unsorted.append((d.name, d\_capacitys\_sum))  
  
 # Сортировка по суммарной ёмкости  
 res\_12 = sorted(res\_12\_unsorted, key=itemgetter(1), reverse=True)  
 print(res\_12)  
  
 print('\nЗадание А3')  
 res\_13 = {}  
 # Перебираем все библиотеки  
 for d in libs:  
 if 'Library' in d.name:  
 # Список дисков в библиотеке  
 d\_disks = list(filter(lambda i: i[2] == d.name, many\_to\_many))  
 # только тип записывающего слоя  
 d\_disks\_id = [x for x, \_, \_ in d\_disks]  
 # Добавляем результат в словарь  
 # ключ - библиотека, значение - список id  
 res\_13[d.name] = d\_disks\_id  
  
 print(res\_13)  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()

**Результаты выполнения:**

****