Московский государственный технический университет имени Н.Э.Баумана

 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Факультет РТ Радиотехнический

Кафедра ИУ5 Системы обработки информации и управления

**Отчет по рубежному контролю № 1 по курсу**

**Базовые компоненты интернет-технологий**

**Вариант 20**

|  |  |  |
| --- | --- | --- |
| Исполнитель |  |  |
| студент группы РТ5-31б | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Титов А.Д. |
|  |  | “\_\_\_” \_\_\_\_\_\_\_\_\_\_ 2021 г. |
|  |  |  |
| Проверил |  |  |
| Доцент кафедры ИУ5 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Гапанюк Ю.Е. |
|  |  | “\_\_\_” \_\_\_\_\_\_\_\_\_\_ 2021 г. |

Москва -  2021

**Описание задания**

1. Необходимо создать два класса данных в соответствии с Вашим вариантом предметной области, которые связаны отношениями один-ко-многим и многие-ко-многим.
2. Необходимо создать списки объектов классов, содержащих тестовые данные (3-5 записей), таким образом, чтобы первичные и вторичные ключи соответствующих записей были связаны по идентификаторам.
3. **Вариант Е.**
   1. «Поставщик» и «Деталь» связаны соотношением один-ко-многим.

Выведите список всех поставщиков, у которых фамилия состоит не менее, чем из 5 букв, и список поставляемых ими деталей.

* 1. «Поставщик» и «Деталь» связаны соотношением один-ко-многим.

Выведите список поставщиков со средней зарплатой, полученной от стоимости продажи деталей, отсортированный по средней зарплате поставщика

* 1. «Поставщик» и «Деталь» связаны соотношением многие-ко-многим. Выведите список всех деталей, у которых название начинается с буквы «М», и фамилии их поставщиков.

**Текст программы**

from functools import \*

class Detail:

"""Деталь"""

def \_\_init\_\_(self, id, name, price, pr\_id):

"""

Args:

id (int): id детали

name (str): название детали

price (int): стоимость детали

pr\_id (int): id поставщика (provider)

"""

self.id = id

self.name = name

self.price = price

self.pr\_id = pr\_id

class Provider:

"""Поставщик"""

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

"""

Args:

id (int): id поставщика

name (str): имя поставщика

"""

self.id = id

self.name = name

class DetPr:

"""

Поставщики деталей (для реализации связи многие-ко-многим)

"""

def \_\_init\_\_(self, pr\_id, det\_id):

self.det\_id = det\_id

self.pr\_id = pr\_id

# Поставщики

providers = [

Provider(1, 'Титов'),

Provider(2, 'Яковенко'),

Provider(3, 'Бения'),

Provider(4, 'Смирнов'),

Provider(5, 'Васильев'),

Provider(6, 'Агеев')

]

# Детали

details = [

Detail(1, 'Процессор', 20000, 1),

Detail(2, 'Материнская плата', 15000, 2),

Detail(3, 'Кулер', 6000, 1),

Detail(4, 'Жесткий диск', 9000, 3),

Detail(5, 'Монитор', 12000, 1)

]

det\_prov = [

DetPr(1,1),

DetPr(2,2),

DetPr(3,3),

DetPr(3,4),

DetPr(3,5),

DetPr(4,1),

DetPr(5,2),

DetPr(6,3),

DetPr(6,4),

DetPr(6,5)

]

def main():

# Соединение данных один-ко-многим

one\_to\_many = [(detail.name, detail.price, provider.name)

for detail in details

for provider in providers

if detail.pr\_id == provider.id]

# Соединение данных многие-ко-многим

many\_to\_many\_temp = [(provider.name, dp.pr\_id, dp.det\_id)

for provider in providers

for dp in det\_prov

if provider.id == dp.pr\_id]

many\_to\_many = [(detail.name, detail.price, provider\_name)

for provider\_name, provider\_id, detail\_id in many\_to\_many\_temp

for detail in details if detail.id == detail\_id]

# Задание Е1

"""

«Поставщик» и «Деталь» связаны соотношением один-ко-многим.

Выведите список всех поставщиков, у которых фамилия состоит не менее, чем из 5 букв, и список поставляемых ими деталей.

"""

print("Задание Е1")

res1 = {}

for item in one\_to\_many:

if (len(item[2]) <= 5):

if (item[2] in res1):

res1[item[2]].append(item[0])

else:

res1[item[2]] = [item[0]]

res1 = list((key, res1[key]) for key in res1.keys())

print(res1)

# Задание Е2

"""

«Поставщик» и «Деталь» связаны соотношением один-ко-многим.

Выведите список поставщиков со средней зарплатой, полученной от стоимости продажи деталей, отсортированный по средней зарплате поставщика.

"""

print("Задание Е2")

res2 = []

for provider in providers:

# список деталей и их стоимость у каждого поставщика

d\_prov = list(filter(lambda i: i[2] == provider.name, one\_to\_many))

if (len(d\_prov) > 0):

res2.append((provider.name, round(reduce(lambda x, y: x + y, [x[1] for x in d\_prov], 0) / len(d\_prov), 2)))

res2 = sorted(res2, key = lambda x: x[1])

print(res2)

"""

«Поставщик» и «Деталь» связаны соотношением многие-ко-многим.

Выведите список всех деталей, у которых название начинается с буквы «М», и фамилии их поставщиков.

"""

print("Задание Е3")

res3 = list(filter(lambda x: x[0][0] == "М", many\_to\_many))

d = {}

for item in res3:

if (item[0] in d):

d[item[0]].append(item[2])

else:

d[item[0]] = [item[2]]

res3 = [(key, d[key]) for key in d.keys()]

print(res3)

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Примеры выполнения программы**

