**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Радиотехнический»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Разработка интернет-приложений»

Рубежный контроль №1

Вариант Е7

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы РТ5-51Б |  | преподаватель каф. ИУ5 |
| Грызин Алексей |  | Гапанюк Ю.Е. |
| Подпись и дата: |  | Подпись и дата: |

# Описание задания

1. «Микропроцессор» и «Компьютер» связаны соотношением один-ко-многим. Выведите список всех Микропроцессоров, у которых в названии присутствует слово «Микропроцессор», и список работающих в них Компьютеров.
2. «Микропроцессор» и «Компьютер» связаны соотношением один-ко-многим. Выведите список Микропроцессоров со средней зарплатой Компьютеров в каждом Микропроцессоре, отсортированный по средней зарплате. Средняя зарплата должна быть округлена до 2 знака после запятой (*Микропроцессорьной функции вычисления среднего значения в Python нет, нужно использовать комбинацию функций вычисления суммы и количества значений; для округления необходимо использовать функцию* [*https://docs.python.org/3/library/functions.html#round*](https://docs.python.org/3/library/functions.html#round)).
3. «Микропроцессор» и «Компьютер» связаны соотношением многие-ко-многим. Выведите список всех Компьютеров, у которых фамилия начинается с буквы «А», и названия их Микропроцессоров.

# Текст программы

models.py

class Processor:

    """Процессор"""

    def \_\_init\_\_(self, id, title, num\_cores, freq, price):

        self.id = id

        self.title = title

        self.numb\_cores = num\_cores

        self.freq = freq

        self.price = price

class PC:

    """Компьютер"""

    def \_\_init\_\_(self, id, title, proc\_id, price):

        self.id = id

        self.title = title

        self.proc\_id = proc\_id

        self.price = price

class ProcessorPC:

    """

    Процессоры компьютеров

    """

    def \_\_init\_\_(self, proc\_id, pc\_id):

        self.proc\_id = proc\_id

        self.pc\_id = pc\_id

mocks.py

from models import \*

processors = [

    Processor(1, 'AMD FX 4300', 4, 3800, 4000),

    Processor(2, 'AMD FX 8300', 8, 3500, 8000),

    Processor(3, 'Микропроцессор AMD Ryzen 7 3800', 8, 3800, 16000),

    Processor(4, 'Intel Core i7-6700', 8, 3800, 16000),

    Processor(5, 'Микропроцессор Intel Core i7-6700K', 8, 4000, 20000),

]

pcs = [

    PC(1, 'BichMachina', 1, 20000),

    PC(2, 'Dota2PC', 5, 50000),

    PC(3, 'NaMinimalkah', 2, 25000),

    PC(4, 'PC4', 3, 35023),

    PC(5, 'PC5', 3, 40123),

    PC(6, 'PC6', 4, 37421),

    PC(7, 'TOP-PC', 5, 52421),

    PC(8, 'BEST-PC', 4, 40000),

    PC(9, 'Античный ПК', 1, 18000),

    PC(10, 'Автобот', 3, 45000),

    PC(11, 'Антивирус', 2, 40322)

]

proc\_pcs = [

    ProcessorPC(1, 1),

    ProcessorPC(1, 9),

    ProcessorPC(2, 3),

    ProcessorPC(2, 11),

    ProcessorPC(3, 4),

    ProcessorPC(3, 5),

    ProcessorPC(3, 10),

    ProcessorPC(4, 6),

    ProcessorPC(4, 8),

    ProcessorPC(5, 2),

    ProcessorPC(5, 7),

]

main.py

from mocks import \*

def **main**():

    one\_to\_many = [(proc.title, proc.id, pc.title, pc.price)

        for proc in processors

        for pc in pcs

        if pc.proc\_id == proc.id]

    many\_to\_many\_temp = [(proc.title, proc\_pc.proc\_id, proc\_pc.pc\_id)

        for proc in processors

        for proc\_pc in proc\_pcs

        if proc.id==proc\_pc.proc\_id]

    many\_to\_many = [(pc.title, title)

        for title, proc\_id, pc\_id in many\_to\_many\_temp

        for pc in pcs if pc.id==pc\_id]

    res\_1 = list(filter(lambda i: i[0].**lower**().**find**('микропроцессор') != -1, one\_to\_many))

    res\_2 = []

    res\_2\_ids = []

    for i in one\_to\_many:

        if res\_2\_ids.**count**(i[1]) != 0:

            continue

        buf = list(filter(lambda x: x[1] == i[1], one\_to\_many))

        res\_2\_ids.**append**(i[1])

        sum = 0

        count = **len**(buf)

        for j in buf:

            sum += j[3]

        res\_2.**append**((i[0], **round**(sum / count, 2)))

    a = ['a', 'а']

    res\_3 = list(filter(lambda i: a.**count**(i[0][0].**lower**()) != 0, many\_to\_many))

**print**('Задание 1', res\_1)

**print**('Задание 2', res\_2)

**print**('Задание 3', res\_3)

if \_\_name\_\_ == "\_\_main\_\_":

**main**()

# Результат работы программы

