РК1

Тенишев Александр, ИУ5-35Б

Вариант Д2.

1. «Класс» и «Школьник» связаны соотношением один-ко-многим. Выведите список всех сотрудников, у которых фамилия заканчивается на «ов», и названия их отделов.
2. «Класс» и «Школьник» связаны соотношением один-ко-многим. Выведите список отделов со средней зарплатой сотрудников в каждом отделе, отсортированный по средней зарплате (*отдельной функции вычисления среднего значения в Python нет, нужно использовать комбинацию функций вычисления суммы и количества значений*).
3. «Класс» и «Школьник» связаны соотношением многие-ко-многим. Выведите список всех отделов, у которых название начинается с буквы «А», и список работающих в них сотрудников.

Результаты выполнения

```
Задание D1
Полное имя:  Иван Иванов        Возраст:  11    Класс 5А
Полное имя:  Мария Петрова      Возраст:  12    Класс 5А
Полное имя:  Анна Кузнецова     Возраст:  13    Класс 7Б
Полное имя:  Спартак Бендеров   Возраст:  17    Класс 10А
Задание D2
Класс:  5А      Средний возраст:  11.5
Класс:  7Б      Средний возраст:  12.666666666666666
Класс:  6В      Средний возраст:  14.0
Класс:  10А     Средний возраст:  17.0
Задание D3
Класс:  5А      Список учеников:
Полное имя:  Иван Иванов        Возраст:  11    Класс 5А
Полное имя:  Мария Петрова      Возраст:  12    Класс 5А
Класс:  10А     Список учеников:
Полное имя:  Спартак Бендеров   Возраст:  17    Класс 10А
```

Текст программы

```python
class Student:
    """Школьник"""
    def __init__(self, student_id, full_name, age, class_id):
        self.student_id = student_id
        self.full_name = full_name
        self.age = age #количественный признак
        self.class_id = class_id #связь один-ко-многим
```

```
    def check_surname(self):
        i = 1
        if (self.full_name[-1] == 'а'):
            i+=1
        if ((self.full_name[-i] == 'в')&(self.full_name[-i-1] == 'о')):
            return True
        return False


class SchoolClass:
    """Класс"""
    def __init__(self, class_id, grade, letter):
        self.class_id = class_id
        self.grade = grade
        self.letter = letter

class Class_Students:
    """Многие-ко-многим"""
    def __init__(self,student_id, class_id):
        self.student_id = student_id
        self.class_id = class_id

"""Функции высших порядков"""
def get_class_name(SchoolClass = [], class_id = 0):
    for x in SchoolClass:
        if (x.class_id==class_id):
            return str(x.grade)+x.letter
    return "NS"

def find_by_surname(Student = [], SchoolClass = []):
    for x in Student:
        if (x.check_surname()==True):
            print("Полное имя: ", x.full_name, "\tВозраст: ", x.age, "\tКласс",
get_class_name(SchoolClass, x.class_id))

def class_by_average_age(Student = [], SchoolClass = []):
    New_list = []
    for x in SchoolClass:
        av_age = 0
        k = 0
        for y in Student:
            if (y.class_id == x.class_id):
                av_age+=y.age
                k+=1
        if (k!=0):
            av_age/=k
        New_list.append(((str(x.grade)+x.letter),av_age))
    New_list = sorted(New_list, key=lambda item: item[1])
    for x in New_list:
        print("Класс: ", x[0], "\tСредний возраст: ", x[1])
```

```python
def find_classes_with_a(Student = [], SchoolClass = [], Relations = []):
    for x in SchoolClass:
        if (x.letter == 'A'):
            print("Класс: ",get_class_name(SchoolClass, x.class_id), "\tСписок
учеников:")
            for y in Relations:
                if y.class_id == x.class_id:
                    for z in Student:
                        if z.student_id == y.student_id:
                            print("Полное имя: ", z.full_name, "\tВозраст: ",
z.age, "\tКласс", get_class_name(SchoolClass, z.class_id))


"""Списки объектов"""
students = [
    Student(student_id=1, full_name="Иван Иванов", age=11, class_id=1),
    Student(student_id=2, full_name="Мария Петрова", age=12, class_id=1),
    Student(student_id=3, full_name="Цекарь Эйсап", age=12, class_id=2),
    Student(student_id=4, full_name="Анна Кузнецова", age=13, class_id=2),
    Student(student_id=5, full_name="Игорь Гофман", age=14, class_id=3),
    Student(student_id=6, full_name="Спартак Бендеров", age=17, class_id=4),
    Student(student_id=7, full_name="Дмитрий Соколовский", age=13, class_id=2)
]

school_classes = [
    SchoolClass(class_id=1, grade=5, letter='А'),
    SchoolClass(class_id=2, grade=7, letter='Б'),
    SchoolClass(class_id=3, grade=6, letter='В'),
    SchoolClass(class_id=4, grade=10, letter='А')
]

class_student_relations = [
    Class_Students(class_id=1, student_id=1),
    Class_Students(class_id=1, student_id=2),
    Class_Students(class_id=2, student_id=3),
    Class_Students(class_id=2, student_id=4),
    Class_Students(class_id=3, student_id=5),
    Class_Students(class_id=4, student_id=6),
    Class_Students(class_id=2, student_id=7)
]


"""Основная программа"""
print("Задание D1")
find_by_surname(students,school_classes) #D1
print("Задание D2")
class_by_average_age(students,school_classes) #D2
print("Задание D3")
find_classes_with_a(students,school_classes,class_student_relations) #D3
```