ИУ5-32Б,

Астахов Иван

***Рубежный контроль 2***

***Вариант 1***

Рубежный контроль представляет собой разработку тестов на языке Python.

1. Проведите рефакторинг текста программы рубежного контроля №1 таким образом, чтобы он был пригоден для модульного тестирования.

class Student:  
 def \_\_init\_\_(self, id, name, age, id\_group):  
 self.id = id  
 self.name = name  
 self.age = age  
 self.id\_group = id\_group

class Group:  
 def \_\_init\_\_(self, id, name, course):  
 self.id = id  
 self.name = name  
 self.course = course

class Student\_Group:  
 def \_\_init\_\_(self, student\_id, group\_id):  
 self.student\_id = student\_id  
 self.group\_id = group\_id

from operator import itemgetter  
from student import Student  
from group import Group  
from student\_group import Student\_Group  
  
def first\_task(one\_to\_many):  
 res\_1 = sorted(one\_to\_many, key=itemgetter(0))  
 return res\_1  
  
def second\_task(one\_to\_many):  
 temp\_dict = {}  
 for student\_name, student\_id\_group, group\_name in one\_to\_many:  
 temp\_dict[group\_name] = temp\_dict.get(group\_name, 0) + 1  
  
 res\_2 = [(group\_name, count) for group\_name, count in temp\_dict.items()]  
 res\_2.sort(key=lambda x: (-x[1], x[0])) # Сортировка по количеству и имени  
 return res\_2  
  
  
def third\_task(many\_to\_many, substring):  
 result = []  
 for student\_name, \_, group\_name in many\_to\_many:  
 if substring in student\_name: # Проверка на наличие подстроки в имени студента  
 result.append((student\_name, group\_name))  
 return result

# main.py  
from student import Student  
from group import Group  
from student\_group import Student\_Group  
from tasks import first\_task, second\_task, third\_task  
  
def main():  
 students = [  
 Student(1, "Emelyanov D.B.", 20, 2),  
 Student(2, "Semenov E.Y", 22, 6),  
 Student(3, "Dmitriev S.A", 21, 17),  
 Student(4, "Pavlenko T.D.", 21, 1),  
 Student(5, "Ivanov P.A.", 20, 3),  
 Student(6, "Petrov I.N.", 21, 1),  
 Student(7, "Sidorov O.K.", 22, 6),  
 Student(8, "Orlov V.M.", 20, 2)  
 ]  
  
 groups = [  
 Group(1, "Group A", 1),  
 Group(2, "Group B", 2),  
 Group(3, "Group C", 3),  
 Group(4, "Group D", 4),  
 Group(5, "Group E", 5)  
 ]  
  
 groups\_students = [  
 Student\_Group(1, 1),  
 Student\_Group(2, 2),  
 Student\_Group(3, 3),  
 Student\_Group(3, 2),  
 Student\_Group(4, 1),  
 Student\_Group(5, 3),  
 Student\_Group(6, 4),  
 Student\_Group(7, 5),  
 Student\_Group(8, 2)  
 ]  
  
 one\_to\_many = [(st.name, st.id\_group, gr.name)  
 for st in students  
 for gr in groups  
 if st.id\_group == gr.id]  
  
 many\_to\_many\_temp = [(st.name, sg.student\_id, gr.name)  
 for st in students  
 for sg in groups\_students  
 if sg.student\_id == st.id]  
  
 many\_to\_many = [(student\_name, student\_id, gr.name)  
 for student\_name, student\_id, group\_id in many\_to\_many\_temp  
 for gr in groups if gr.id == group\_id]  
  
 print('Задание 1')  
 print(first\_task(one\_to\_many))  
  
 print("\nЗадание 2")  
 print(second\_task(one\_to\_many))  
  
 print("\nЗадание 3")  
 print(third\_task(many\_to\_many, 'ov'))  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()

1. Для текста программы рубежного контроля №1 создайте модульные тесты с применением TDD - фреймворка (3 теста).

import unittest

from tasks import first\_task, second\_task, third\_task

class TestTasks(unittest.TestCase):

def test\_first\_task(self):

one\_to\_many = [("Emelyanov D.B.", 2, "Group B"), ("Semenov E.Y", 6, "Group F")]

self.assertEqual(first\_task(one\_to\_many), [("Emelyanov D.B.", 2, "Group B"), ("Semenov E.Y", 6, "Group F")])

def test\_second\_task(self):

one\_to\_many = [("Emelyanov D.B.", 2, "Group B"), ("Semenov E.Y", 6, "Group F"), ("Ivanov P.A.", 3, "Group C")]

self.assertEqual(second\_task(one\_to\_many), [("Group B", 1), ("Group C", 1), ("Group F", 1)])

def test\_third\_task(self):

many\_to\_many = [("Emelyanov D.B.", 1, "Group A"), ("Semenov E.Y", 2, "Group B"), ("Orlov V.M.", 8, "Group H")]

self.assertEqual(third\_task(many\_to\_many, 'em'), [('Semenov E.Y', 'Group B')])

if \_\_name\_\_ == '\_\_main\_\_':

unittest.main()

Результаты тестирования:  
