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.

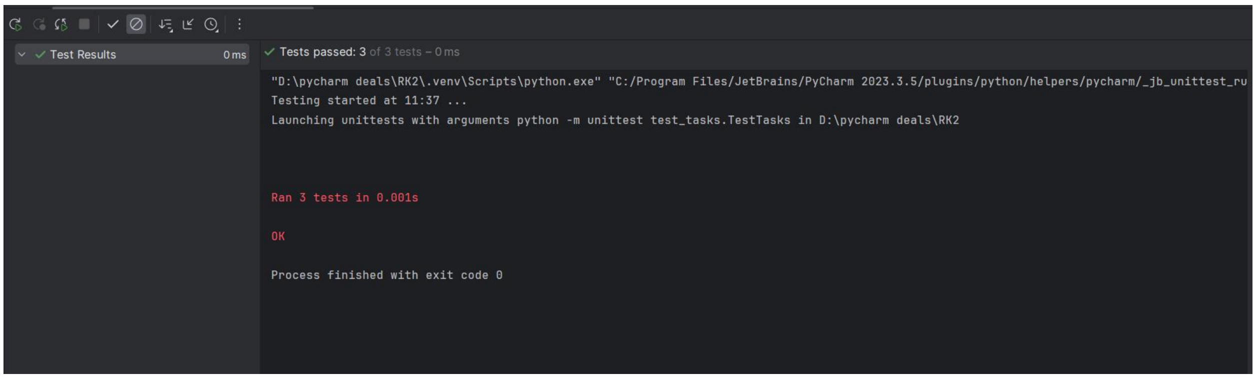
```
class Student:
    def __init__(self, id, name, age, id_group):
        self.id = id
        self.name = name
        self.age = age
        self.id_group = id_group

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()
```

2) 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:



Tests passed: 3 of 3 tests – 0 ms

```
"D:\pycharm deals\RK2\.venv\Scripts\python.exe" "C:/Program Files/JetBrains/PyCharm 2023.3.5/plugins/python/helpers/pycharm/_jb_unittest_ru
Testing started at 11:37 ...
Launching unittests with arguments python -m unittest test_tasks.TestTasks in D:\pycharm deals\RK2



Ran 3 tests in 0.001s

OK

Process finished with exit code 0
```