**Министерство науки и высшего образования Российской Федерации** **Федеральное государственное бюджетное образовательное учреждение высшего образования** **«Московский государственный технический университет** **имени Н.Э. Баумана**

**(национальный исследовательский университет)»**

**(МГТУ им. Н.Э. Баумана)**

**Факультет «Информатика и системы управления»**

**Кафедра ИУ5 «Системы обработки информации и управления»**

Рубежный контроль №1

по дисциплине «Базовые компоненты интернет технологий»

Выполнил:

студент группы ИУ5-33Б

Лукьянов К.Е.

Проверил:

Юрий Гапанюк

2022 г.

Программа тестов:

import unittest  
from main import Computer, Program, ProgramComp, task\_1, task\_2, task\_3  
  
  
class TestDB(unittest.TestCase):  
 def setUp(self):  
 # Компьютеры  
 self.computers = [  
 Computer(1, 'Plastic white frame'),  
 Computer(2, 'Metal pink frame'),  
 Computer(3, 'flint yellow frame'),  
  
 Computer(11, 'Metal black frame'),  
 Computer(22, 'Metal red frame'),  
 Computer(33, 'Plastic black frame'),  
 ]  
  
 # Програмы  
 self.programs = [  
 Program(1, 'C++', 1000, 1),  
 Program(2, 'Python', 500, 2),  
 Program(3, 'C', 400, 3),  
 Program(4, 'C++', 2000, 3),  
 Program(5, 'Python', 100, 3),  
 ]  
  
 self.programs\_comps = [  
 ProgramComp(1, 1),  
 ProgramComp(2, 2),  
 ProgramComp(3, 3),  
 ProgramComp(3, 4),  
 ProgramComp(3, 5),  
  
 ProgramComp(11, 1),  
 ProgramComp(22, 2),  
 ProgramComp(33, 3),  
 ProgramComp(33, 4),  
 ProgramComp(33, 5),  
 ]  
 # Соединение данных один-ко-многим  
 self.one\_to\_many = [(program.language, program.length, computer.frame)  
 for program in self.programs  
 for computer in self.computers  
 if program.computer\_id == computer.id]  
  
 # Соединение данных многие-ко-многим  
 many\_to\_many\_temp = [(computer.frame, pr\_c.computer\_id, pr\_c.program\_id)  
 for computer in self.computers  
 for pr\_c in self.programs\_comps  
 if computer.id == pr\_c.computer\_id]  
 self.many\_to\_many = [(program.language, program.length, frame)  
 for frame, computer\_id, program\_id in many\_to\_many\_temp  
 for program in self.programs  
 if program.id == program\_id]  
  
 def test\_task\_1(self):  
 result = set(task\_1(self.one\_to\_many))  
 answer = {('Python', 'Metal pink frame'),  
 ('Python', 'flint yellow frame')}  
 self.assertEqual(answer, result)  
  
 def test\_task\_2(self):  
 result = task\_2(self.one\_to\_many)  
 answer = [('flint yellow frame', 100),  
 ('Metal pink frame', 500),  
 ('Plastic white frame', 1000)]  
 self.assertEqual(answer, result)  
  
 def test\_task\_3(self):  
 result = task\_3(self.many\_to\_many)  
 answer = [('Python', 100, 'flint yellow frame'),  
 ('Python', 100, 'Plastic black frame'),  
 ('C', 400, 'flint yellow frame'),  
 ('C', 400, 'Plastic black frame'),  
 ('Python', 500, 'Metal pink frame'),  
 ('Python', 500, 'Metal red frame'),  
 ('C++', 1000, 'Plastic white frame'),  
 ('C++', 1000, 'Metal black frame'),  
 ('C++', 2000, 'flint yellow frame'),  
 ('C++', 2000, 'Plastic black frame')]  
 self.assertEqual(answer, result)  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 unittest.main()

Изменённая программа рк1:

from operator import itemgetter  
from math import inf  
  
  
class Program:  
  
 def \_\_init\_\_(self, id, language, length, computer\_id):  
 self.id = id  
 self.language = language  
 self.length = length  
 self.computer\_id = computer\_id  
  
  
class Computer:  
  
 def \_\_init\_\_(self, id, frame):  
 self.id = id  
 self.frame = frame  
  
  
class ProgramComp:  
  
 def \_\_init\_\_(self, computer\_id, program\_id):  
 self.computer\_id = computer\_id  
 self.program\_id = program\_id  
  
  
def task\_1(one\_to\_many):  
 *"""«Программа» и «Компьютер» связаны соотношением один-ко-многим. Выведите список всех программ, Ккоторые \  
 написаны на языке Python, и корпусов компьютера."""* return [(record[0], record[2]) for record in one\_to\_many if record[0].startswith('Python')]  
  
  
def task\_2(one\_to\_many):  
 *"""«Программа» и «Компьютер» связаны соотношением один - ко - многим.Выведите список компьютеров с минимальной \  
 длинной программ кода, отсортированный по минимальной длинне."""* mins = {}  
 for language, length, processor in one\_to\_many:  
 mins[processor] = min(mins.get(processor, inf), length)  
 return sorted(mins.items(), key=itemgetter(1))  
  
  
def task\_3(many\_to\_many):  
 *"""«Программа» и «Компьютер» связаны соотношением многие-ко-многим. Выведите список всех связанных программ и компьютеров,\  
 отсортированный по программам, сортировка по компьютерам произвольная. """* return sorted(many\_to\_many, key=itemgetter(1))

Итог программы:

