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, искусственный интеллект и системы управления»

Кафедра «Системы обработки информации и управления»

Рубежный контроль №2 по курсу «Базовые компоненты интернет-технологий»

Выполнил:

студент группы ИУ5-33Б

Попов Степан

Проверил:

Доцент кафедры ИУ5

Гапанюк Юрий Евгеньевич

2022 г.

Текст программы

CD.py (реализация классов)

```
class CD_Disk:
   """CD-disk"""
   def __init__(self, id, name, size, libr_id):
      self.id = id
      self.name = name
      self.size = size  # MB size
      self.libr_id = libr_id


class CD_Library:
   """CD-library"""
   def __init__(self, id, street):
      self.id = id
      self.street = street


class Disk_Libr:
   def __init__(self, libr_id, disk_id):
      self.libr_id = libr_id
      self.disk_id = disk_id
```

main.py (реализация связей и функций)

```
from operator import itemgetter
from CD import *


# 1:M
def one_disk_to_many_libraries(disks, libraries):
   return [(d.name, d.size, l.street)
      for d in disks
      for l in libraries
```

```
        if d.libr_id==l.id]


# M:M
def many_disks_to_many_libraries(disks, libraries, disks_librs):
    many_to_many_temp = [(l.street, dl.disk_id)
        for l in libraries
        for dl in disks_librs
        if l.id==dl.libr_id]

    return [(d.name, d.size, lib_street)
        for lib_street, disk_id in many_to_many_temp
        for d in disks if d.id==disk_id]

# Задание B1: Названия дисков, начинающихся с N
def disks_names_from_N(one_to_many):
    return list(filter(lambda iter: iter[0][0] == 'N', one_to_many))

# Задание B2: список библиотек с диском минимального размера в каждой библиотеке,
# отсортированный по размеру диска
def librs_with_minimal_disk_size_sorted(one_to_many):
    dict_dynamic = {street: size for _,size,street in one_to_many}
    for _,size,street in one_to_many:
        if size < dict_dynamic[street]:
            dict_dynamic[street] = size
    return sorted(dict_dynamic.items(), key=itemgetter(1))

# Задание B3: список всех связанных дисков и библиотек, отсортированный по названию дисков
def librs_to_disks_sorted_by_name(many_to_many):
    return sorted(many_to_many, key = itemgetter(0, 2))
```

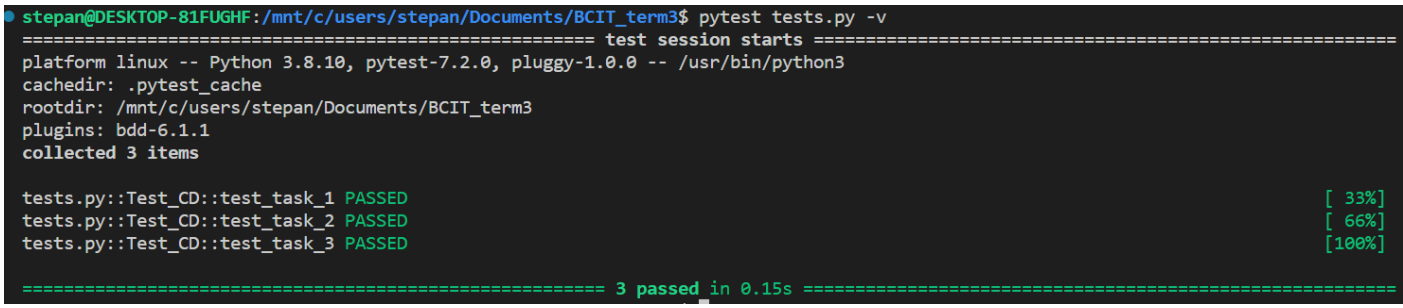
tests.py (тестирование)

```
import unittest
from CD import *
from main import *


class Test_CD(unittest.TestCase):
    def setUp(self):
        self.libraries = [
            CD_Library(1, 'Братиславская'),
            CD_Library(2, 'Проспект мира'),
            CD_Library(3, 'Верхние поля'),

            CD_Library(11, 'Невский проспект'),
            CD_Library(22, 'Гороховая'),
            CD_Library(33, 'Академическая')
        ]

        self.disks = [
            CD_Disk(1, 'Nirvana', 2000, 1),
            CD_Disk(2, 'Drain Gang', 3500, 2),
            CD_Disk(3, 'No counrty for old man', 4500, 3),
            CD_Disk(4, 'Reservoir Dogs', 2500, 3),
            CD_Disk(5, 'Trainspotting', 3500, 3)
        ]

        self.disks_librs = [
            Disk_Libr(1,1),
            Disk_Libr(2,2),
            Disk_Libr(3,3),
            Disk_Libr(3,4),
            Disk_Libr(3,5),
```

```
            Disk_Libr(11,1),
            Disk_Libr(22,2),
            Disk_Libr(33,3),
            Disk_Libr(33,4),
            Disk_Libr(33,5)
        ]

        self.one_to_many = one_disk_to_many_libraries(self.disks, self.libraries)
        self.many_to_many = many_disks_to_many_libraries(self.disks, self.libraries, self.disks_librs)

    def test_task_1(self):
        supposed_res = [
            ('Nirvana', 2000, 'Братиславская'),
            ('No counrty for old man', 4500, 'Верхние поля')
        ]
        res = disks_names_from_N(self.one_to_many)
        self.assertEqual(res, supposed_res)

    def test_task_2(self):
        supposed_res = [
            ('Братиславская', 2000),
            ('Верхние поля', 2500),
            ('Проспект мира', 3500)
        ]
        res = librs_with_minimal_disk_size_sorted(self.one_to_many)
        self.assertEqual(res, supposed_res)

    def test_task_3(self):
        supposed_res = [
            ('Drain Gang', 3500, 'Гороховая'),
            ('Drain Gang', 3500, 'Проспект мира'),
            ('Nirvana', 2000, 'Братиславская'),
```

```
            ('Nirvana', 2000, 'Невский проспект'),
            ('No counrty for old man', 4500, 'Академическая'),
            ('No counrty for old man', 4500, 'Верхние поля'),
            ('Reservoir Dogs', 2500, 'Академическая'),
            ('Reservoir Dogs', 2500, 'Верхние поля'),
            ('Trainspotting', 3500, 'Академическая'),
            ('Trainspotting', 3500, 'Верхние поля')
        ]
        res = librs_to_disks_sorted_by_name(self.many_to_many)
        self.assertEqual(res, supposed_res)

if __name__ == '__main__':
    unittest.main()
```

Результат


stepan@DESKTOP-81FUGHF:/mnt/c/users/stepan/Documents/BCIT_term3$ pytest tests.py -v
test session starts
platform linux -- Python 3.8.10, pytest-7.2.0, pluggy-1.0.0 -- /usr/bin/python3
cachedir: .pytest_cache
rootdir: /mnt/c/users/stepan/Documents/BCIT_term3
plugins: bdd-6.1.1
collected 3 items
tests.py::Test_CD::test_task_1 PASSED [33%]
tests.py::Test_CD::test_task_2 PASSED [66%]
tests.py::Test_CD::test_task_3 PASSED [100%]
3 passed in 0.15s