Московский государственный технический университет им. Н.Э. Баумана

Факультет ИУ
Кафедра ИУ5

Курс «Основы информатики»
Отчет по Рубежному контролю №2

Выполнил студент группы ИУ5-33Б:
Громов В.С.
Подпись и дата:

Проверил преподаватель каф.:
Гапанюк Ю. Е.
Подпись и дата:

Москва, 2024 г

Постановка задачи

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.

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

Текст программы

```
import unittest
from collections import Counter

class DB:
        def __init__(self, ID: int, name: str, descr: str):
        self.ID = ID
        self.descr = descr
        self.name = name

class Proc:
        def __init__(self, proc_id: int, name: str, db_id: int):
        self.proc_id = proc_id
        self.name = name
        self.db_id = db_id

class ProcDB:
        def __init__(self, proc_id: int, db_id: int):
        self.proc_id = proc_id
        self.db_id = db_id

def get_one_to_many(dbs, procs):
        return [(proc.name, db.name) for db in dbs for proc in procs if proc.db_id == db.ID]

def get_many_to_many(procs_db, procs, dbs):
        db_dict = {db.ID: db.name for db in dbs}
        return [(proc.name, db_dict[procdb.db_id]) for procdb in procs_db for proc in procs if proc.proc_id
== procdb.proc_id]

def count_procs_per_db(dbs, procs_db):
        db_proc_count = Counter(procdb.db_id for procdb in procs_db)
        return [(db.name, db_proc_count.get(db.ID, 0)) for db in dbs]



procs = [
Proc(1, "create_user", 1),
Proc(2, "delete_user", 1),
Proc(3, "describe_db(mean, median, 25%, 50%, 75%, min, max)", 3),
Proc(4, "get_user_by_id", 3),
Proc(5, "get_log", 2),
```

```
Proc(6, "update_info_user", 1)
]
DBs = [
DB(1, "main_db", "основная бд"),
DB(2, "log_db", "бд для логов"),
DB(3, "ml_db", "бд для создания модели машинного обучения"),
DB(34, "main_other_db", "основная бд (другая)"),
DB(35, "log_other_db", "бд для логов (другая)"),
DB(36, "ml_other_db", "бд для мл (другая)")
]
procs_db = [
ProcDB(1, 1),
ProcDB(2, 1),
ProcDB(3, 1),
ProcDB(4, 3),
ProcDB(5, 3),
ProcDB(1, 34),
ProcDB(2, 35),
ProcDB(3, 36),
ProcDB(4, 35),
ProcDB(2, 35)
]
```

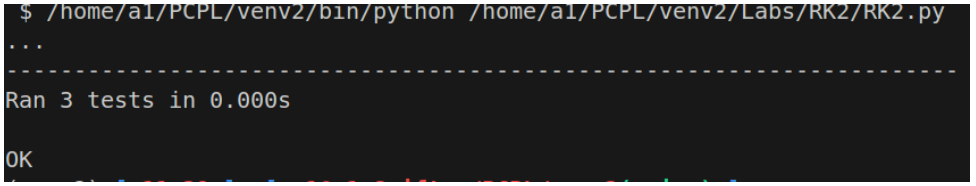
```
class Test(unittest.TestCase):
        def test_one_to_many(self):
        dbs = [DB(1, "db1", "descr1")]
        procs = [Proc(1, "proc1", 1)]
        self.assertEqual(get_one_to_many(dbs, procs), [('proc1', 'db1')])

def test_many_to_many(self):
        procs_db = [ProcDB(1,1)]
        procs = [Proc(1, "proc1", 1)]
        dbs = [DB(1, "db1", "descr1")]
        self.assertEqual(get_many_to_many(procs_db, procs, dbs), [('proc1', 'db1')])

def test_count_procs_per_db(self):
        dbs = [DB(1,"db1","descr1"), DB(2,"db2","descr2")]
        procs_db = [ProcDB(1,1),ProcDB(2,1)]
        self.assertEqual(count_procs_per_db(dbs, procs_db), [('db1', 2), ('db2', 0)])


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

Анализ результатов


$ /home/al/PCPL/venv2/bin/python /home/al/PCPL/venv2/Labs/RK2/RK2.py
...
Ran 3 tests in 0.000s
OK