Тянутов Александр ИУ5-31Б

РК 1

Вариант А.
1. «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список всех связанных сотрудников и отделов, отсортированный по отделам, сортировка по сотрудникам произвольная.
2. «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список отделов с суммарной зарплатой сотрудников в каждом отделе, отсортированный по суммарной зарплате.
3. «Отдел» и «Сотрудник» связаны соотношением многие-ко-многим. Выведите список всех отделов, у которых в названии присутствует слово «отдел», и список работающих в них сотрудников.

Переделанное задание под заданные классы:
Классы библиотек и языков программирования rk_name_lib rk_name_pl
1. Вывести отсортированный список по языкам программирования
2. Вывести сумму веса бинарных файлов в библиотеке
3. Вывести имя библиотек, содержащих name(задается в check_str), и использующихся языков программирования относящиеся к этой библиотеке

rk_library.py

```
import itertools

class rk_name_lib:
    newid = itertools.count()
    def __init__(self, name : str, lst_pl = []): # ubdi - unit byte digital information\
        self.id = next(self.newid)
        self.name = name
        self.lst_pl = lst_pl
    def get_id(self):
        return self.id
    def __repr__(self) -> str:
        return "id: {} name: {} | lst = {}\n".format(self.id, self.name, self.lst_pl)
```

prog_langs.py

```
import itertools

class rk_name_pl:
    newid = itertools.count()
    def __init__(self, name, weight):
        self.id = next(self.newid)
        self.name = name
        self.weight_binaries = weight
    def get_id(self):
        return self.id
    def get_weight(self):
        return self.weight_binaries
```

```
    def __repr__(self) -> str:
        return "id: {} name: {}\n".format(self.id, self.name)
```

main.py

```
from rk_library import rk_name_lib
from prog_lang import rk_name_pl

class Combined_table:
    def __init__(self, id_pl : rk_name_pl, id_lib : rk_name_lib):
        self.id_lib = id_lib.get_id()
        self.id_pl = id_pl.get_id()
    def __lt__(self, other):
        return self.id_pl < other.id_pl
    def __repr__(self) -> str:
        return "{}->{}".format(self.id_pl, self.id_lib)


def task_1(fct: list[Combined_table]):
    print('Before sorting\n', fct)
    fct.sort(reverse=True)
    print('After sorting\n', fct)

def task_2(fct: list[Combined_table], lst_pl: list[rk_name_pl], lst_libs: list[rk_name_lib]):
    t = 0
    temp_dct = {}
    for conn in fct:
        if temp_dct.get(conn.id_lib) is None:
            temp_dct[conn.id_lib] = []
        temp_dct[conn.id_lib].append(conn.id_pl)
    values_fct = {}
    for k, v in temp_dct.items():
        temp_summ = 0
        for vv in v:
            for i in lst_pl:
                if vv == i.get_id():
                    temp_summ += i.get_weight()
        values_fct[k] = temp_summ
    for k, v in values_fct.items():
        for i in lst_libs:
            if k == i.get_id():
                print(i.name, ' = ', v)

def task_3(fct: list[Combined_table], lst_pl: list[rk_name_pl], lst_libs: list[rk_name_lib],
check_str):
    dct_libs_langs = {}
    for i in lst_libs:
        if check_str in i.name:
            dct_libs_langs[i.get_id()] = []
```

```
            for item in fct:
                if item.id_lib == i.get_id():
                    dct_libs_langs[i.get_id()].append(item.id_pl)
    if not bool(dct_libs_langs):
        print('Nothing to print')
        return
    for k, v in dct_libs_langs.items():
        for i in lst_libs:
            if k == i.get_id():
                lib = i
        pl_ans = []
        for vv in v:
            for j in lst_pl:
                if vv == j.get_id():
                    pl_ans.append(j.name)
        print(lib.name, ' = ', pl_ans)

if __name__ == "__main__":
    lst_rk_l = []
    lst_rk_pl = []
    rk_l1 = rk_name_pl('FirstPl', 100)
    lst_rk_pl.append(rk_l1)
    rk_l2 = rk_name_pl('SecondPl', 200)
    lst_rk_pl.append(rk_l2)
    rk_l3 = rk_name_pl('ThirdPl', 300)
    lst_rk_pl.append(rk_l3)
    lib_1 = rk_name_lib('FirstLib', [rk_l1])
    lst_rk_l.append(lib_1)
    lib_2 = rk_name_lib('SecondLib', [rk_l1, rk_l2])
    lst_rk_l.append(lib_2)
    lib_3 = rk_name_lib('ThirdLib', [rk_l1])
    lst_rk_l.append(lib_3)

    libs = [
        lib_1,
        lib_2,
        lib_3
    ]
    prog_langs = [
        rk_l1,
        rk_l2,
        rk_l3
    ]


    libs_langs = [
        Combined_table(rk_l1, lib_1),
        Combined_table(rk_l1, lib_2),
        Combined_table(rk_l2, lib_2),
```

```
    Combined_table(rk_l1, lib_3)
  ]

  print('='*50, '\nA1 | Sort by programming language id')
  task_1(libs_langs)
  print('='*50, '\nA2 | Print out summ of programming language binaries for each library')
  task_2(libs_langs, prog_langs, libs)
  check_str = 'S'
  print('='*50, '\nA3 | Print out names of libraries contains \'{}\' in name and language that
they are conatining '.format(check_str))
  task_3(libs_langs, prog_langs, libs, check_str)
  print('='*50)
```