

МГТУ им. Баумана

Факультет «Информатика и системы управления»

Кафедра «Системы обработки информации и управления»

Курс «Разработка интернет-приложений»

Отчет по рубежному контролю №1

Вариант Г 12

Выполнил: Кравцов Андрей, ИУ5-54Б

Преподаватель: Ю.Е. Гапанюк

Москва, 2021 г.

## **Условия рубежного контроля №1 по курсу РИП**

Рубежный контроль представляет собой разработку программы на языке Python, которая выполняет следующие действия:

1) Необходимо создать два класса данных в соответствии с Вашим вариантом предметной области, которые связаны отношениями один-ко-многим и многие-ко-многим.

Пример классов данных для предметной области Сотрудник-Отдел:

1. Класс «Сотрудник», содержащий поля:
  - ID записи о сотруднике;
  - Фамилия сотрудника;
  - Зарплата (количественный признак);
  - ID записи об отделе. (для реализации связи один-ко-многим)
2. Класс «Отдел», содержащий поля:
  - ID записи об отделе;
  - Наименование отдела.
3. (Для реализации связи многие-ко-многим) Класс «Сотрудники отдела», содержащий поля:
  - ID записи о сотруднике;
  - ID записи об отделе.

2) Необходимо создать списки объектов классов, содержащих тестовые данные (3-5 записей), таким образом, чтобы первичные и вторичные ключи соответствующих записей были связаны по идентификаторам.

3) Необходимо разработать запросы в соответствии с Вашим вариантом. Запросы сформулированы в терминах классов «Сотрудник» и «Отдел», которые используются в примере. Вам нужно перенести эти требования в Ваш вариант предметной области. При разработке запросов необходимо по возможности использовать функциональные возможности языка Python (list/dict comprehensions, функции высших порядков).

Для реализации запроса №2 введите в класс, находящийся на стороне связи «много», произвольный количественный признак, например, «зарплата сотрудника».

Результатом рубежного контроля является документ в формате PDF, который содержит текст программы и результаты ее выполнения.

12	Язык программирования	Средство разработки
----	--------------------------	---------------------

## Текст программы.

```
# -*- coding: utf-8 -*-

# используется для сортировки
from operator import itemgetter

# используется для красивого вывода
from prettytable import PrettyTable


class PLanguage:
    """Язык программирования"""

    def __init__(self, id, name, year, popularity, ide_id):
        self.id = id
        self.name = name
        self.year = year
        self.popularity = popularity # 0 to 100 more points = more popular
        self.ide_id = ide_id


class IDE:
    """Средство разработки"""

    def __init__(self, id, name):
        self.id = id
        self.name = name


class PLanguageIDE:
```

```
"""Реализации связи многие-ко-многим для среды и языка"""
```

```
def __init__(self, pl_id, ide_id):
```

```
    self.pl_id = pl_id
```

```
    self.ide_id = ide_id
```

```
IDEs = [
```

```
    IDE(1, 'VS Code'),
```

```
    IDE(2, 'PyCharm'),
```

```
    IDE(3, 'GoLand'),
```

```
    IDE(4, 'Vim'),
```

```
    IDE(5, 'Xcode'),
```

```
]
```

```
PLanguages = [
```

```
    PLanguage(1, 'C++', 1983, 92, 1),
```

```
    PLanguage(2, 'JavaScript', 1995, 88, 1),
```

```
    PLanguage(3, 'Scala', 2004, 55, 1),
```

```
    PLanguage(4, 'Python', 1991, 100, 2),
```

```
    PLanguage(5, 'Go', 2009, 78, 3),
```

```
    PLanguage(6, 'C', 1972, 94, 4),
```

```
    PLanguage(7, 'BASIC', 1964, 55, 4),
```

```
    PLanguage(8, 'Lisp', 1958, 30, 4),
```

```
    PLanguage(9, 'Assembly', 1949, 63,4),
```

```
    PLanguage(10, 'SWIFT', 2014, 70, 5),
```

```
    PLanguage(11, 'Ruby', 1995, 63, 5),
```

```
]
```

```
PLanguageIDEs = [
```

```
    PLanguageIDE(1, 1),
```

```
    PLanguageIDE(2, 1),
```

```
PLanguageIDE(3, 1),
PLanguageIDE(4, 1),
PLanguageIDE(5, 1),
PLanguageIDE(6, 1),
PLanguageIDE(7, 1),
PLanguageIDE(8, 1),
PLanguageIDE(9, 1),
PLanguageIDE(10, 1),
PLanguageIDE(11, 1),
PLanguageIDE(4, 2),
PLanguageIDE(5, 3),
PLanguageIDE(1, 4),
PLanguageIDE(2, 4),
PLanguageIDE(4, 4),
PLanguageIDE(6, 4),
PLanguageIDE(7, 4),
PLanguageIDE(10, 5),
PLanguageIDE(11, 5),
PLanguageIDE(1, 5),
PLanguageIDE(2, 5),
PLanguageIDE(5, 5),
]
```

```
def getPlsPopularityById(id):
```

```
    popularity = [(pl.popularity)
```

```
    for pl in PLanguages
```

```
    if pl.ide_id == id]
```

```
    return max(popularity)
```

```
def main():
```

```
# Соединение данных один-ко-многим
```

```

one_to_many = [(ide.id, ide.name, pl.popularity)
for ide in IDEs
for pl in PLanguages
if (pl.ide_id == ide.id)]

# Соединение данных многие-ко-многим
many_to_many_temp = [(ide.name, pi.ide_id, pi.pl_id)
for ide in IDEs
for pi in PLanguageIDEs
if ide.id == pi.ide_id]

many_to_many = [(pl.name, pl.year, pl.popularity, ide_name)
for ide_name, ide_id, pl_id in many_to_many_temp
for pl in PLanguages if pl.id == pl_id]

print('Задание Г1')
"""
Выведите список всех IDE, у которых название начинается с буквы «V»,
и список соответствующих им языков.
"""

task1 = [(pl.name, pl.year, pl.popularity, ide.name)
for ide in IDEs
for pl in PLanguages
if (pl.ide_id == ide.id)&(ide.name[0] == "V")]

table = PrettyTable()
table.field_names = ["Prog. Language", "Year", "Popularity", "IDE"]
table.add_rows(task1)
print(table)

```

```
print('\nЗадание Г2')
```

```
"""
```

Выведите список IDE с максимальной популярностью языков в каждой IDE,  
отсортированный по максимальной популярности.

```
"""
```

```
table = PrettyTable()
```

```
table.field_names = ["IDE", "Popularity"]
```

```
task2 = []
```

```
for ide in IDEs:
```

```
PlsofIde= list(filter(lambda i: i[0] == ide.id, one_to_many))
```

```
PlofIdeMaxPopular = list(filter(lambda i: i[2] == getPlsPopularityById(ID(ide.id), PlsofIde))[0])
```

```
task2.append(PlofIdeMaxPopular[1:])
```

```
table.add_rows(sorted(task2, key=lambda i: i[1], reverse=True))
```

```
print(table)
```

```
print('\nЗадание Г3')
```

```
"""
```

Выведите список всех связанных языков и IDE, отсортированный по IDE,  
сортировка по языкам произвольная.

```
"""
```

```
res_13 = sorted(many_to_many, key=itemgetter(2), reverse=True)
```

```
table = PrettyTable()
```

```
table.field_names = ["Prog. Language", "Year", "Popularity", "IDE"]
```

```
table.add_rows(res_13)
```

```
print(table)
```

```
if __name__ == '__main__':
```

```
main()
```

## Результаты работы программы.

Год	1990	98	75-8000
1990	1990	1990	1990