Московский Государственный Технический Университет им. Н. Э. Баумана

Факультет «Информатика и Системы управления»

Кафедра «Автоматизированные системы обработки информации и управления»

Дисциплина «Разработка Интернет-Приложений»

**Отчёт по рубежному контролю №1**

**Вариант 21**

Выполнил:

Студент группы ИУ5-53Б

**Ткачева Д.А.**

Преподаватель:

**Гапанюк Ю.Е.**

**Москва, 2020 г.**

1. **Задание**

|  |  |  |
| --- | --- | --- |
| № варианта | Класс 1 | Класс 2 |
| 21 | Оператор | Язык программирования |

Вариант В

1 «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите

список всех сотрудников, у которых фамилия начинается с буквы «А», и

названия их отделов.

2 «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите

список отделов с минимальной зарплатой сотрудников в каждом отделе,

отсортированный по минимальной зарплате.

3 «Отдел» и «Сотрудник» связаны соотношением многие-ко-многим. Выведите

список всех связанных сотрудников и отделов, отсортированный по

сотрудникам, сортировка по отделам произвольная.

1. **Текст программы**

*# используется для сортировки*from operator import itemgetter  
  
  
class Opr:  
 *"""Операторы"""* def \_\_init\_\_(self, id, name, priority, prg\_id):  
 self.id = id  
 self.name = name  
 self.priority = priority  
 self.prg\_id = prg\_id  
  
 def \_\_str\_\_(self):  
 return **f"**{self.id}**,** {self.name}**,** {self.priority}**"**class Prg:  
 *"""Язык программирования"""* def \_\_init\_\_(self, id, name):  
 self.id = id  
 self.name = name  
  
  
class OprPrg:  
 *"""  
 'Операторы языка' для реализации  
 связи многие-ко-многим  
 """* def \_\_init\_\_(self, pr\_id, op\_id):  
 self.pr\_id = pr\_id  
 self.op\_id = op\_id  
  
  
*# Операторы*oprs = [  
 Opr(1, **'+ сложение'**, 4, 1),  
 Opr(2, **'/ деление'**, 3, 5),  
 Opr(3, **'\* умножение'**, 3, 3),  
 Opr(4, **'>= сравнение'**, 6, 2),  
 Opr(5, **'<= сравнение'**, 6, 2),  
]  
  
*# Языки*prgs = [  
 Prg(1, **'Python'**),  
 Prg(2, **'C++'**),  
 Prg(3, **'Go'**),  
 Prg(4, **'Java'**),  
 Prg(5, **'C#'**),  
 Prg(6, **'Pascal'**),  
]  
  
oprs\_prg = [  
 OprPrg(1, 1),  
 OprPrg(2, 5),  
 OprPrg(3, 3),  
 OprPrg(4, 2),  
 OprPrg(5, 2),  
  
 OprPrg(2, 1),  
 OprPrg(6, 2),  
 OprPrg(4, 3),  
 OprPrg(1, 4),  
 OprPrg(5, 5),  
]  
  
  
def main():  
 *"""Основная функция"""  
  
 # Соединение данных один-ко-многим only start C* one\_to\_many = [{k.name: [o.name for o in oprs if o.prg\_id == k.id]}  
 for k in prgs  
 if k.name[0] == **'C'**]  
 o\_t\_m = {k.name: [o for o in oprs if o.prg\_id == k.id] for k in prgs}  
  
 *# Соединение данных многие-ко-многим* many\_to\_many = [(p.name, op.pr\_id, op.op\_id)  
 for p in prgs  
 for op in oprs\_prg  
 if p.id == op.pr\_id]  
  
  
  
 print(**'Задание B1'**)  
 print(one\_to\_many)  
  
 print(**'**\n**Задание B2'**)  
 res\_2 = []  
  
 for kv in o\_t\_m:  
 mn = 100  
 for o in o\_t\_m[kv]:  
 if o.priority < mn:  
 mn = o.priority  
 if mn < 100:  
 res\_2.append((kv, mn))  
  
 res\_12 = sorted(res\_2, key=itemgetter(1), reverse=True)  
 print(res\_12)  
  
 print(**'**\n**Задание B3'**)  
 res\_13 = {}  
  
 for d in prgs:  
 un\_sort = list(filter(lambda i: i.prg\_id == d.id, oprs))  
 res\_13[d.name] = sorted(un\_sort, key=lambda item: getattr(item, **"priority"**), reverse=True)  
  
 for pr in res\_13:  
 print(**"{0}: {1}"**.format(pr, [item.\_\_str\_\_() for item in res\_13[pr]]))  
  
  
if \_\_name\_\_ == **'\_\_main\_\_'**:  
 main()

1. **Результат выполнения программы**

