# Отчет по рубежному контролю №1 по курсу базовые компоненты интернет-технологий (БКИТ)

|  |  |
| --- | --- |
| ИСПОЛНИТЕЛЬ: |  |
| студент группы ИУ5Ц-54Б Клеша Егор |  |
|  | (подпись) |
| Гапанюк Ю.Е. | " " 2022 г. |

Москва, МГТУ - 2022

# СОДЕРЖАНИЕ ОТЧЕТА

1. [Листинг программы: 2](#_bookmark0)
2. [Результаты работы программы 5](#_bookmark1)

# Листинг программы:

from operator import itemgetter  
'''  
Вариант Г  
  
«Язык программирования» и «Синтаксическая конструкция» связаны соотношением один-ко-многим.   
Выведите список всех языков программирования, у которых название начинается с буквы «П», и список содержащих в них синтаксической конструкции.  
  
«Язык программирования» и «Синтаксическая конструкция» связаны соотношением один-ко-многим.   
Выведите список языков программирования с максимальной сложностью синтаксических конструкции в каждом языке программирования, отсортированный по максимальной сложности.  
  
«Язык программирования» и «Синтаксическая конструкция» связаны соотношением многие-ко-многим.   
Выведите список всех связанных языков программирования и синтаксических конструкций, отсортированный по языками, сортировка по синтаксической конструкции произвольная.   
'''  
  
# класс - Синтаксическая конструкция  
class Syntactic\_construction:  
 def \_\_init\_\_(*self*, *id*, *name*, *complexity*, *Programming\_language\_id*):  
 # номер синтаксического конструкции  
 *self*.id = *id* # название синтаксического конструкции  
 *self*.name = *name* # сложность на которой находится синтаксическая конструкция  
 *self*.complexity = *complexity* # номер языков программирования  
 *self*.Programming\_language\_id = *Programming\_language\_id*# класс - Язык программирования  
class Programming\_language:  
 def \_\_init\_\_(*self*, *id*, *name*):  
 *self*.id = *id  
 self*.name = *name*# классы - Язык программирования и Синтаксическая конструкция  
class Syntactic\_construction\_Programming\_language:  
 def \_\_init\_\_(*self*, *Programming\_language\_id*, *Syntactic\_construction\_id*):  
 *self*.Programming\_language\_id = *Programming\_language\_id  
 self*.Syntactic\_construction\_id = *Syntactic\_construction\_id*Programming\_language\_s = [  
 Programming\_language(1, 'Джава скрипт'),  
 Programming\_language(2, 'Пхп'),  
 Programming\_language(3, 'Питон'),  
 Programming\_language(4, 'Свифт'),  
 Programming\_language(5, 'Kотлин')  
]  
Syntactic\_construction\_s = [  
 Syntactic\_construction(1, 'Легко', 10, 1),  
 Syntactic\_construction(2, 'Немножко среднее', 20, 3),  
 Syntactic\_construction(3, 'Среднее', 30, 2),  
 Syntactic\_construction(4, 'Немножко сложно', 40, 4),  
 Syntactic\_construction(5, 'Сложно', 50, 5),  
 Syntactic\_construction(6, 'Сверхсложно', 60, 1)  
]  
Syntactic\_construction\_s\_of\_Programming\_language\_s = [  
 Syntactic\_construction\_Programming\_language(3, 1),  
 Syntactic\_construction\_Programming\_language(4, 2),  
 Syntactic\_construction\_Programming\_language(5, 3),  
 Syntactic\_construction\_Programming\_language(1, 4),  
 Syntactic\_construction\_Programming\_language(2, 5),  
 Syntactic\_construction\_Programming\_language(6, 6),  
]  
  
def main():  
 one\_to\_many = [(ch.name, ch.complexity, Programming\_language.name)  
 for Programming\_language in Programming\_language\_s  
 for ch in Syntactic\_construction\_s  
 if ch.Programming\_language\_id == Programming\_language.id]  
  
 many\_to\_many\_temp = [(Programming\_language.name, ChOfProgramming\_language\_s.Programming\_language\_id, ChOfProgramming\_language\_s.Syntactic\_construction\_id)  
 for Programming\_language in Programming\_language\_s  
 for ChOfProgramming\_language\_s in Syntactic\_construction\_s\_of\_Programming\_language\_s  
 if Programming\_language.id == ChOfProgramming\_language\_s.Programming\_language\_id]  
  
 many\_to\_many = [(ch.name, ch.complexity, Programming\_language\_name)  
 for Programming\_language\_name, Programming\_language\_id, ch\_id in many\_to\_many\_temp  
 for ch in Syntactic\_construction\_s if ch.id == ch\_id]  
  
 print('')  
 print('Задание Г\_1 - начинается с П')  
 array\_dict = {}  
 for lib\_name, x, Programming\_language\_name in one\_to\_many:  
 # если название языков программирования начинается с 'А'  
 if Programming\_language\_name[0] == 'П':  
 if Programming\_language\_name in array\_dict:  
 array\_dict[Programming\_language\_name].append(lib\_name)  
 else:  
 array\_dict[Programming\_language\_name] = [lib\_name]  
 print(\*array\_dict.items())  
  
 print('')  
 print('Задание Г\_2 - номер по максимальной сложности')  
 array\_dict\_2 = {}  
 for x, func\_num, Programming\_language\_name in one\_to\_many:  
 if Programming\_language\_name in array\_dict\_2:  
 array\_dict\_2[Programming\_language\_name] = max(array\_dict\_2[Programming\_language\_name], func\_num)  
 else:  
 array\_dict\_2[Programming\_language\_name] = func\_num  
 array\_dict\_2 = {key: value for key, value in sorted(array\_dict\_2.items(), key=lambda *item*: item[1])}  
 print(\*array\_dict\_2.items())  
  
 print('')  
 print('Задание Г\_3 - отсортированный по языками, сортировка по синтаксической конструкции произвольная')  
 array\_list = []  
 for lib\_name, x, Programming\_language\_name in many\_to\_many:  
 array\_list.append((Programming\_language\_name, lib\_name))  
 array\_list = sorted(array\_list, key=lambda *item*: item[0])  
 print(\*array\_list)  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()

# Результаты работы программы:

Задание Г\_1 - начинается с П

('Пхп', ['Среднее']) ('Питон', ['Немножко среднее'])

Задание Г\_2 - номер по максимальной сложности

('Питон', 20) ('Пхп', 30) ('Свифт', 40) ('Kотлин', 50) ('Джава скрипт', 60)

Задание Г\_3 - отсортированный по языками, сортировка по синтаксической конструкции произвольная

('Kотлин', 'Среднее') ('Джава скрипт', 'Немножко сложно') ('Питон', 'Легко') ('Пхп', 'Сложно') ('Свифт', 'Немножко среднее')

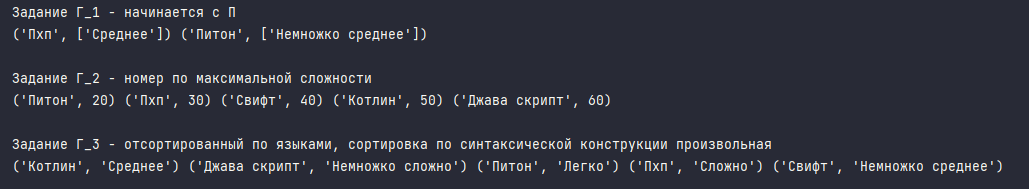


Рисунок 1 - Скриншот результатов работы программы