'''Вариант - E, вариант предметной области - 23

("Синтаксическая конструкция - язык программирования")'''

class SynCon:

# Синтаксическая конструкция

def \_\_init\_\_(self, id, name, len\_con, lan\_id):

self.id = id

self.name = name

self.len\_con = len\_con # кол-во символов в названии конструкции

self.lan\_id = lan\_id

class LanProg:

# Язык программирования

def \_\_init\_\_(self, id, name):

self.id = id

self.name = name

class SynLan:

# 'Синтаксические конструкция языка программирования' для реализации

# связи многие-ко-многим

def \_\_init\_\_(self, lan\_id, syn\_id):

self.lan\_id = lan\_id

self.syn\_id = syn\_id

# языки программирования

lans = [

LanProg(1, 'C'),

LanProg(2, 'C#'),

LanProg(3, 'C++'),

LanProg(4, 'Python'),

]

# синтаксические конструкции

syns = [

SynCon(1, 'if', 2, 2),

SynCon(2, 'else', 4, 2),

SynCon(3, 'while', 5, 4),

SynCon(4, 'for', 3, 4),

SynCon(5, 'switch', 6, 3),

SynCon(6, 'case', 4, 3),

SynCon(6, 'elif', 4, 1)

]

syns\_lans = [

SynLan(1, 1),

SynLan(1, 2),

SynLan(1, 3),

SynLan(1, 4),

SynLan(1, 5),

SynLan(1, 6),

SynLan(2, 1),

SynLan(2, 2),

SynLan(3, 1),

SynLan(3, 2),

SynLan(3, 3),

SynLan(3, 4),

SynLan(3, 5),

SynLan(3, 6),

SynLan(4, 1),

SynLan(4, 2),

SynLan(4, 3),

SynLan(4, 4),

]

def main():

# Основная функция

# Соединение данных один-ко-многим

one\_to\_many = [(s.name, s.len\_con, l.name)

for l in lans

for s in syns

if s.lan\_id == l.id]

# Соединение данных многие-ко-многим

many\_to\_many\_temp = [(l.name, sy.lan\_id, sy.syn\_id)

for l in lans

for sy in syns\_lans

if l.id == sy.lan\_id]

many\_to\_many = [(s.name, s.len\_con, lan\_name)

for lan\_name, lan\_id, syn\_id in many\_to\_many\_temp

for s in syns if s.id == syn\_id]

print('Задание E1')

# выбираем языки, в названии которых есть 'C'

res\_1 = list(filter(lambda x: 'C' in x[2], one\_to\_many))

were = ''

for elem in res\_1:

if(elem[2] == were):

print(' ', elem[0], elem[1])

else:

were = elem[2]

print(were, ":", sep='')

print(' ', elem[0], elem[1])

print('\nЗадание E2')

avg\_len = dict()

for link in one\_to\_many:

if (link[2] in avg\_len):

avg\_len[link[2]].append(link[1])

else:

avg\_len[link[2]] = [link[1]]

for key, value in avg\_len.items():

print(key, round(sum(value) / len(value), 2))

print('\nЗадание E3')

# выбираем синтаксические единицы, которые начинаются с 'e'

res\_3 = list(filter(lambda x: x[0][0] == 'e', many\_to\_many))

for i in range(len(res\_3)):

print(str(i + 1) + '.', res\_3[i][0], res\_3[i][2])

if \_\_name\_\_ == '\_\_main\_\_':

main()

