Домашняя работа 1 модуль 3 урок

<<Функции>>

Основы функций

Задача 1

Условие:

Программист Виталий, вдохновленный новыми знаниями о функциональном программировании, решил оптимизировать свою телефонную книгу. Он хочет написать функцию, которая будет преобразовывать контакты в нужный формат. Каждый контакт имеет вид "Фамилия Имя Отчество" или "Фамилия Имя", если у человека отсутствует отчество.
Помогите Виталию реализовать функцию get_initials(), которая принимает 3 или 2 аргумента (ФИО или ФИ) и возвращает инициалы в формате "Фамилия И. О." или "Фамилия И.".

Входные данные:

В одной строке вводятся через пробел ФИО или ФИ контакта в телефоне.

Выходные данные:

В одной строке выведите инициалы контакта.

Примеры:

Пример	Ввод	Вывод
1	Ярцев Валерий Рустэмович	Ярцев В.Р.
2	Иванов Иван Иванович	Иванов И.И.
3	Фантикова Елизавета	Фантикова Е.

Подсказки:

1) Для того чтобы функция принимала 2 или 3 аргумента, задайте отчество как аргумент по умолчанию (например, `surname=""`)
2) Теперь в функции вы можете проверить есть ли отчество через условный оператор if
3) Удобно подавать в функцию `*(input().split())`. Таким образом у вас подастся либо 3 аргумента, либо 2 аргумента

Задача 2

Условие:

Виталий выражает свою огромную благодарность за вашу предыдущую программу! Тем не менее он обнаружил, что у некоторых его иностранных друзей полные имена состоят из более чем трех слов. Поэтому Виталий просит вас внести изменения в программу, чтобы она могла строить инициалы для всех друзей, включая иностранных.

Входные данные:
В одной строке через пробел вводится полное имя контакта Виталия.

Выходные данные:
В одной строке выведите инициалы контакта.

Примеры:

Пример	Ввод	Вывод
1	Li Zhang Xiao Mei	Li Z.X.M.
2	Müller Schmidt Hans Peter	Müller S.H.P.
3	Abdulaziz Al Ahmed Mahmood	Abdulaziz A.A.M.

Подсказки:

1) Принимайте в функции позиционные аргументы, как `def func(*args):`
2) Подавайте в функцию *(input().split()). Таким образом у вас подастся либо 3 аргумента, либо 2 аргумента

Задача 3
Условие:
Геннадий, программист из престижной IT-компании "АвиаМозги", обратился к вам с важной задачей. Компания решила подарить бесплатные лицензии на свои продукты всем студентам, зарегистрированным в образовательной базе. Геннадий доверил вам написать функцию для выполнения этой задачи, учитывая следующую информацию:

У вас есть список студентов со следующей структурой:

```
students = [
   {"id": 367673, "full_name": "Ярцев Валерий Рустэмович"},
   {"id": 563234, "full_name": "Шиптенко Виталий Программирович"},
   {"id": 982123, "full_name": "Датабейзов Иван Джетлагович"},
]
```

Вам необходимо реализовать функцию give_license(student_id) и подробно задокументировать ее.

Входные данные:
В функцию подается 1 аргумент: идентификационный номер студента

Выходные данные:
Функция `give_license(student)` должна вернуть bool значение (True - студенту можно выдать лицензию (он есть в образовательной базе), False - студенту не полагается лицензия (так как его нет в базе))

Примеры:
Как и полагается истинному руководителю, Геннадий не сочел нужным предоставить Вам пример работы функции, Вы должны руководствоваться здравой логикой и предоставленным ТЗ.

Подсказки:

1) Для объявления глобальной переменной используйте `global students`

Рекурсия

Задача 4
Условие:
Вы решили пересмотреть свои карьерные предпочтения и искать новую работу, так как вы не согласны с политикой компании "АвиаМозги", которая ограничивает доступ студентов к образовательным продуктам. Вы начали отправлять свое резюме и проходить собеседования, и одно из них - с корпорацией счастья "Вугол". Во время технического собеседования вам задали простое задание по числам Фибоначчи. Вам нужно реализовать рекурсивную функцию, которая будет подсчитывать k-ое число в последовательности чисел Фибоначчи.

Входные данные:
В первой строке подается целое число n - количество чисел, которые Вам надо вычислить
В следующих n строках подаются целые числа k (k >= 1) - порядковые номера чисел в последовательности чисел Фибоначчи.

Выходные данные:
В одной строке выведите целое число - k-ое число в последовательности Фибоначчи.

Примеры:

Пример	Ввод	Вывод
1	3 1 2 3	0 1 1
2	1 13	144
3	3 20	4181

	30 35	514229 5702887

Подсказки:

1) Используйте рекурсию!
2) Вам нужно задать 2 базовых случая - начальные числа в последовательности

Декораторы

Задача 5

Условие:

Отличная работа! Однако чтобы превзойти остальных кандидатов, вам необходимо оптимизировать код.

Интервьюер, которому вы сразу понравились, решил вам немного помочь. Заметим, что для вычисления 35-ого числа в последовательности Фибоначчи необходимо вычислить все предыдущие числа. Так, если мы уже посчитали 30-ое число, то можем использовать его значение при рекурсивном вызове для вычисления 35-ого числа.

Для ускорения работы программы вы можете написать декоратор, который будет хранить подсчитанные значения в словаре. Например, при вызове функции с аргументом 10 мы сохраняли бы вычисленное значение в словаре, как {10: 34}. При следующем вызове функции с аргументом 11 мы могли бы проверить, есть ли в словаре уже вычисленное значение для 10 (поскольку рекурсивный вызов 11 включает вызовы 9 и 10), и использовать его, не рассчитывая заново.

Ваша задача состоит в написании декоратора для функции, реализующей вычисление чисел Фибоначчи, согласно условию этой задачи.

Входные данные:

В первой строке подается целое число n - количество чисел, которые Вам надо вычислить.

В следующих n строках подаются целые числа k (k >= 1) - порядковые номера чисел в последовательности чисел Фибоначчи.

Выходные данные:

В одной строке выведите целое число - k-ое число в последовательности Фибоначчи.

Примеры:

Пример	Ввод	Вывод
1	3 1 2 3	0 1 1
2	1 13	144

3	3 20 30 35	4181 514229 5702887

Подсказки:

1) Используйте стандартный вид декоратора:

```
def my_decorator():
    def real_decorator(func):
        def wrapper(*args, **kwargs):
            result = func(*args, **kwargs)
            return result
        return wrapper
    return real_decorator
```

2) Декоратор нужно использовать как @my_decorator() над функцией из предыдущей задачи