

Примеры заданий для языков ФП

Язык Haskell

Задание 1. Написать программу для нахождения n -го члена последовательности, заданной следующей рекуррентной формулой.

$$a_0 = 1; a_1 = 2;$$

$$a_n = 3 * a_{n-1} - 2 * a_{n-2} + 1 \text{ при } n = 2, 3, \dots$$

Имейте в виду, что прямое программирование данной формулы «как есть» приводит к крайне неэффективной программе!

Задание 2. Совершенным числом называется натуральное число, равное сумме всех своих делителей, включая единицу, но исключая само это число. Так, например, число 28 – совершенное, поскольку $28 = 1 + 2 + 4 + 7 + 14$. Написать программу для нахождения первых n совершенных чисел.

Задание 3. Близнецами называется пара натуральных чисел, каждое из которых равно сумме делителей другого числа. Так, например, числа 220 и 284 – близнецы (проверьте!). Написать программу для нахождения первых n пар близнецов.

Pytho

```
import random
names = ["Маша", "Петя", "Вася"]
secret_names = map(lambda x: random.choice(["Шпунтик", "Винтик", "Фунтик"]), names)
print secret_names
```

Упражнение 1 . Попробуйте переписать следующий код через map. Он принимает список реальных имён и заменяет их прозвищами, используя более надёжный метод.

```
Names = ["Маша", "Петя", "Вася"]
for i in range(len(names)):
    names[i] = hash(names[i])
print names # =>
```

Упражнение 2 : перепишите следующий код, используя map, reduce и filter. Filter принимает функцию и коллекцию. Возвращает коллекцию тех вещей, для которых функция возвращает True.

```
People = [{"имя": "Маша", "рост": 160}, {"имя": "Саша", "рост": 80}, {"имя": "Паша"}]
height_total = 0
height_count = 0
for person in people:
    if "рост" in person:
        height_total += person["рост"]
        height_count += 1
if height_count > 0:
    average_height = height_total / height_count
print average_height # => 120
```

Обмен значений численных переменных

Пользователь вводит два числа. Одно присваивается одной переменной, а второе - другой. Необходимо поменять значения переменных так, чтобы значение первой оказалось во второй, а второй - в первой.

Clojure

Задача 1

Задание

Найти сумму всех чисел от 1 до 4000 999, которые делятся на 3 или 5.

Алгоритм

1. Создать последовательность чисел от 1 до 4000-999 включительно.
2. Отфильтровать её, оставив только нужные нам числа.
3. Просуммировать все числа.

Задача 2

Задание

Найти сумму всех чётных чисел Фиббоначи меньших 4 миллионов.

Алгоритм

1. Строим последовательность Фиббоначи `fib-seq` (бесконечную, благодаря ленивой инициализации).
2. Строим вторую последовательность из чисел Фиббоначи меньших 4 миллионов.
3. Отбираем только чётные.
3. Суммируем.

Задача 3

Задание

Найти максимальный простой делитель числа 600851475143.

Алгоритм

У нас будет 2 функции.

Одна вспомогательная, которая делит число на максимальную степень его делителя, чтобы «избавиться» от делителя в числе.

Вторая основная, которая поочерёдно пробегает по всем простым числам и пытается наше число на них делить. И соответственно то простое число, после которого мы получили 1 и будет искомым.