**ЛАБОРАТОРНАЯ РАБОТА №1**

**ОСНОВНЫЕ ОПЕРАТОРЫ И КОНСТРУКЦИИ PYTHON**

**Вариант 1. *Кролики и рекуррентные отношения***

**Условие задачи:** В 1202 году Леонардо Пизанский (более известный как Фибоначчи) исследовал задачу размножения кроликов в популяции. При этом он пользовался следующими предположениями:

**1**. Популяция начинается на первом месяце с пары новорождённых кроликов.

**2**. Кролики достигают репродуктивного возраста спустя месяц после рождения.

**3**. Каждый месяц каждый кролик репродуктивного возраста спаривается с

другим взрослым кроликом.

**4**. Ровно через месяц после спаривания каждая пара рожает крольчонка и

крольчиху.

**5**. Кролики не умирают и не прекращают размножаться.

Задачей Фибоначчи было вычислить, сколько будет пар кроликов через год. Можно заметить, что на 2-й месяц два кролика достигают репродуктивного возраста и спариваются. На 3-й месяц рождается новая пара кроликов, т. е. популяция состоит уже из двух пар кроликов; первая пара опять спаривается. На 4-й месяц у первой пары кроликов рождается новая пара кроликов, в то время как вторая пара кроликов достигает половозрелого возраста (т. е. теперь уже три пары). Динамика численности популяции представлена на рисунке 1. Спустя год популяция состоит из 144 пар кроликов.

**Последовательность** – *упорядоченное множество объектов (обычно чисел), которые, вообще говоря, могут повторяться.* Последовательности бывают конечные и бесконечные. Будем использовать обозначение an(а энное) для n-ого члена последовательности.

**Рекуррентное отношение** – *это способ задания членов последовательности с учётом значений предыдущих членов.* В случае с кроликами Фибоначчи каждому месяцу соответствует число кроликов, равное количеству кроликов в предыдущем месяце плюс новое потомство. Заметим, что число новорождённых кроликов в любом месяце равно количеству кроликов два месяца назад. В итоге имеем, что если обозначить через Fn число пар кроликов на n-ый месяц, то получим последовательность Фибоначчи, в которой члены определяются по формуле Fn = Fn-1 + Fn-2 (при этом последовательность начинается с членов F1=F2=1).

Найдя n-ый член последовательности, определённой рекуррентным отношением, мы можем просто использовать ту же формулу для вычисления дальнейших элементов последовательности. Эта задача является простейшим примером ***динамического программирования***, которое успешно вычисляет результаты той или иной задачи, используя ответы, полученные для более мелких случаев.

**Входные данные**: натуральные числа n ≤ 40 и k ≤ 5.

**Выходные данные**: общее число кроличьих пар на n-ый месяц, если в 1-ый месяц есть всего одна пара, а у каждой пары половозрелых кроликов рождается помёт из k пар кроликов (вместо одной).

***Ход решения:***

**1.Написание алгоритма**

**2.Написание кода**

**3.Проверка кода**

**4.Всё готово!**

**Анализ задачи**

Фух, начнём наш анализ задачи. На первый взгляд может показаться что ничего не понятно, потому что много условий сразу забивают голову. Разложим нашу задачу на подпункты для лучшего понимания, что от нас требуется.

Нам нужно написать программу для подсчёта кроликов, которые рождаются и, достигая репродуктивного возраста, рождают других кроликов. В задаче у нас есть условие, что количество рождённых кроликов задаёт юзер. Тогда попробуем представить работу программы.

Допустим, что количество рождаемых кроликов 1й парой составляет 1 пару крольчат. Тогда на 2й месяц мы будем иметь значение крольчат 1 пара репродуктивного возраста, и 1 пара не репродуктивного возраста. На 3й месяц мы будет иметь 1 пару кроликов не репродуктивного, и 2 репродуктивного возраста. Если продолжать это дальше, то мы просто придём к ряду Фибоначчи 1 2 3 5 8 ….

Всё будет меняться если юзер захочет ввести 2 в параметр, отвечающий за кол-во рождающихся кроликов 1й парой. Тогда мы будем иметь следующее:

1й месяц- 2 пары не репродуктивного и 1 репродуктивного. Общее кол-во: 3

2й месяц- 2 пары не репродуктивного и 3 репродуктивного. Общее кол-во: 5

3й месяц – 6 пар не репродуктивного и 5 репродуктивного Общее кол-во: 11

4й месяц – 10 пар не репродуктивного и 11 репродуктивного. Общее кол-во: 21

5й месяц – 22 пары не репродуктивного и 21 репродуктивного. Общее кол-во: 43

Здесь мы можем заметить, что число репродуктивных пар в следующем месяце численно равно количеству общих пар в предыдущем месяце. А что же с числом не репродуктивных? Здесь все просто! Число не репродуктивных кроликов численно равно количество репродуктивных особей в прошлом месяце умноженные на количество крольчат, которое 1 пара репродуктивных кроликов должна родить (параметр вводится юзером). Вот и всё. Всё что нужно у нас есть. Осталось всё это перевести в код!

**Написание кода**

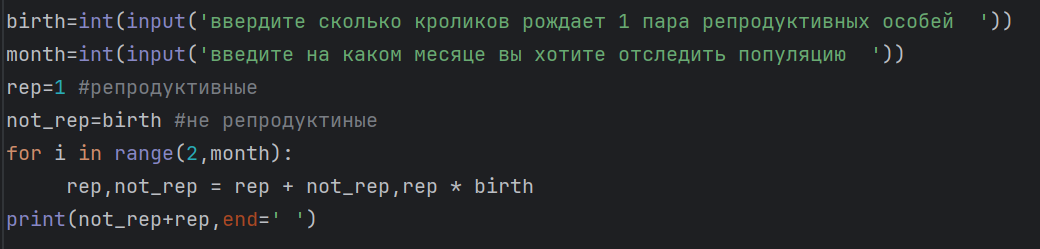
Возьмём кол-во репродуктивных особей за переменную **rep**(reproductive), кол-во не репродуктивных особей за **not\_rep** (not reproductive),количество всех кроликов за **all,** кол-во месяцев за **month,** помёт 1й пары за **birth.**

Программа не представляет из себя ничего сложного. Весь код можно уместить в 1 цикл for.

for i in range(1,month):  
 rep,not\_rep = rep + not\_rep,rep \* birth

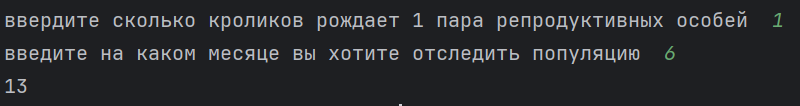
Вот и всё! Осталось только ввести изначальные значение rep=1 и not\_rep=0. Выводим ответ: print(not\_rep+rep,end=' '). Проверим код.

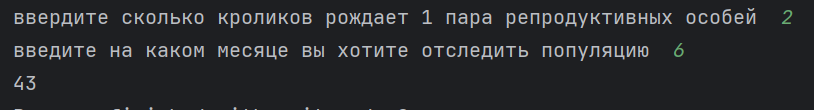
При вводе в birth = 1 и проверяем разные значения month всё идёт гладко. Но когда мы меняем birth на 2 или другое значение выходит что-то странное. Это происходит потому, что значение not\_rep у нас = 0 и что бы мы там дальше не писали это не имеет значение, любое число при обнажении на 0 получается 0. Это очень легко исправить введя в значение not\_rep значение birth (потому что значение не репродуктивных особей равно количеству помёта, которое 1 пара репродуктивных кроликов способна родить). Конечная версия кода выглядит так:

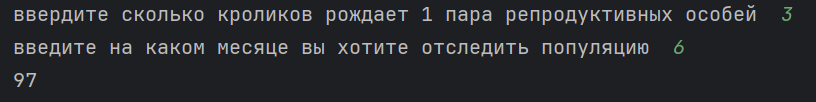


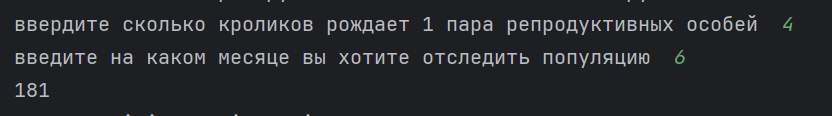
birth=int(input('ввердите сколько кроликов рождает 1 пара репродуктивных особей '))  
month=int(input('введите на каком месяце вы хотите отследить популяцию '))  
rep=1 #репродуктивные  
not\_rep=birth #не репродуктиные  
for i in range(2,month):  
 rep,not\_rep = rep + not\_rep,rep \* birth  
print(not\_rep+rep,end=' ')

**Проверка кода**

****

****

****

****

**Всё готово!**