**Лабораторная работа 2**

**Проблемы Ханойской башни**

**Описание проблемы**

Это классическая проблема, которая иллюстрирует, как рекурсия в сочетании с стратегией «разделяй и властвуй» может позволить тебе тривиально решить проблему, которая кажется очень сложной.   
Суть проблемы состоит в следующем. У вас есть три стержня, на одном из которых расположены N дисков разных размеров. Диски имеют размеры 1, 2, 3, ... N и лежат меньшее на большем. Задача состоит в том, чтобы перенести пирамиду из дисков за наименьшее число ходов на другой стержень. За один раз разрешается переносить только одно кольцо, причём нельзя класть большее кольцо на меньшее. Диски не должны размещаться на стержнях в последовательном порядке, то есть ормально иметь диск 5 поверх диска 10 (но не поверх диска 4).

Алгоритм решения этой проблемы выглядит следующим образом. Мы назовем наши стержни peg A, peg B, и peg C. Предположим, вы хотите переместить верхние N дисков с стержня A на B.

1. Сначала обработайте базовый случай, то есть тот случай, который можно решить немедленно. Здесь базовый случай - это когда перемещается только один диск (N = 1). В этом случае вы просто переместите диск.
2. Если N> 1, процесс выглядит следующим образом:
   1. Переместите верхние диски N-1 от А до C.
   2. Переместите диск N с А на В.
   3. Переместите верхние диски N-1 из C в B.

Обратите внимание, что шаги 1 и 3 предполагают рекурсивный вызов этого алгоритма. Мы называем это рекурсивным случаем.

Оба этих случая должны обрабатываться в одной и той же функции (методе).

Удивительно, но это все. Рекурсивные алгоритмы могут показаться магическими, но их можно понять по аналогии с математической индукцией. С помощью математической индукции вы хотите доказать, что что-то истинно для всех чисел N> 1. Для этого вы докажите, что это верно для N = 0, а затем вы покажете, что, если известно, что это верно для N-1, может быть верно и для N. Учитывая это, вы можете показать, что это верно для любого N> 0. Здесь мы видим, что мы можем переместить 1 диск (N = 1). Если мы знаем, что можем перемещать N-1 диски, то ясно, что мы можем перемещать и N дисков (это рекурсивный случай).

Еще одна полезная вещь - сделать небольшое значение N (скажем, 3) и проработать этот алгоритм на карандаше и бумаге до тех пор, пока вы не убедите себя, что он работает.

**Программа python**

Напишите свою программу в одном файле "hanoi.py" который будет вызываться следующим образом:

python hanoi.py <n>

Где <n> - количество дисков, которые начинаются с первого стержня. Если число не указано, выведите сообщение об ошибке.

Поскольку это не автономный скрипт, не добавляйте

#! /usr/bin/env python

в верхней части файла, и не делайте его исполняемым. Это касается всех будущих лаб.

Ваша программа должна определить класс Hanoi, который инкапсулирует все переменные. В частности, вы должны иметь явное представление о том, какие диски находятся на каких стержнях в любой момент времени. Самый простой способ сделать это - привязать стержни к списку python, а для дисков иметь номера, которые хранятся в списках в правильном порядке. Ваш класс Hanoi должен содержать следующие методы:

* display, который печатает текущее состояние проблемы (какие диски находятся на стержнях).
* solve, что решает проблему. Решение не должно принимать никаких аргументов.
* move, перемещающий один диск с одного стержня на другой. Не забудьте сделать проверку.

Вы, вероятно, захотите также определить другие методы (например, вспомогательный метод для solve () - hint, hint).

Кроме того, перед началом и после каждого перемещения вы должны выводить состояние проблемы (используя свой метод отображения). Дисплей должен выглядеть точно так (для 8 дисков):

0: 4 1

1: 8 7 6 5

2: 3 2

Где 0:, 1:, 2: идентифицируют стержни , а другие номера идентифицируют диски. Убедитесь, что сообщения разделены хотя бы одной пустой строкой.

Ваш метод \_\_init\_\_ должен принимать общее количество дисков в качестве единственного аргумента (кроме self, конечно) и проверять, что это число является положительным целым числом. Вызвать исключение ValueError, если это не так, вместе с сообщением об ошибке.

**Дополнительное задание**

* Добавить код для подсчета количества ходов при решении проблемы. Выведите этот номер после того, как вы решите проблему. Как это число зависит от количества дисков?
* Добавить код, чтобы определить максимальную глубину рекурсии, достигнутую при решении проблемы и выведите ее. Как это число зависит от количества дисков?
* Добавьте параметр командной строки -v (для «подробного») который, если присутствует, выведет состояние проблемы после каждого перемещения. Если его нет, проблема будет решена беззвучно. разрешить -v быть расположенным где угодно в командной строке (если только после аргументов python hanoi.py, конечно).