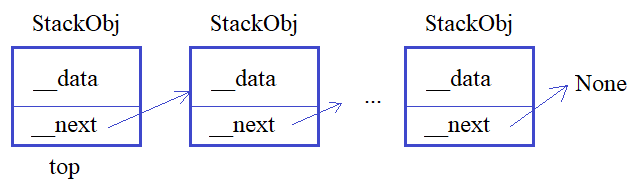
Теория по односвязным спискам (при необходимости): <https://youtu.be/TrHAcHGIdgQ>

**Подвиг 7.** Используя информацию о модуле abc из предыдущего подвига 6, объявите базовый класс с именем **StackInterface** со следующими абстрактными методами:

def push\_back(self, obj) - добавление объекта в конец стека;  
def pop\_back(self) - удаление последнего объекта из стека.



На основе этого класса объявите дочерний класс с именем Stack. Объекты этого класса должны создаваться командой:

st = Stack()

и в каждом объекте этого класса должен формироваться локальный атрибут:

\_top - ссылка на первый объект стека (для пустого стека \_top = None).

В самом классе Stack переопределить абстрактные методы базового класса:

def push\_back(self, obj) - добавление объекта в конец стека;  
def pop\_back(self) - удаление последнего объекта из стека.

Сами объекты стека должны определяться классом StackObj и создаваться командой:

obj = StackObj(data)

где data - информация, хранящаяся в объекте (строка). В каждом объекте класса StackObj должны автоматически формироваться атрибуты:

\_data - информация, хранящаяся в объекте (строка);  
\_next - ссылка на следующий объект стека (если следующий отсутствует, то \_next = None).

Пример использования классов (эти строчки в программе писать не нужно):

st = Stack()

st.push\_back(StackObj("obj 1"))

obj = StackObj("obj 2")

st.push\_back(obj)

del\_obj = st.pop\_back() # del\_obj - ссылка на удаленный объект (если объектов не было, то del\_obj = None)

P.S. В программе требуется объявить только классы. На экран выводить ничего не нужно.

from abc import ABC, abstractmethod

# здесь объявляйте классы