1.4 Пространства имён и области видимости 10 из 10 шагов пройдено 13 из 13 баллов получено

Реализуйте программу, которая будет эмулировать работу с пространствами имен. Необходимо реализовать поддержку создания пространств имен и добавление в них переменных.

В данной задаче у каждого пространства имен есть уникальный текстовый идентификатор – его имя.

Вашей программе на вход подаются следующие запросы:

- **create <namespace> <parent>** – создать новое пространство имен с именем **<namespace>** внутри пространства **<parent>**
- **add <namespace> <var>** – добавить в пространство **<namespace>** переменную **<var>**
- **get <namespace> <var>** – получить имя пространства, из которого будет взята переменная **<var>** при запросе из пространства **<namespace>**, или **None**, если такого пространства не существует

Рассмотрим набор запросов

- **add global a**
- **create foo global**
- **add foo b**
- **create bar foo**
- **add bar a**

Структура пространств имен описанная данными запросами будет эквивалентна структуре пространств имен, созданной при выполнении данного кода

```
a = 0
def foo():
  b = 1
  def bar():
    a = 2
```

В основном теле программы мы объявляем переменную **a**, тем самым добавляя ее в пространство **global**. Далее мы объявляем функцию **foo**, что влечет за собой создание локального для нее пространства имен внутри пространства **global**. В нашем случае, это описывается командой **create foo global**. Далее мы объявляем внутри функции **foo** функцию **bar**, тем самым создавая пространство **bar** внутри пространства **foo**, и добавляем в **bar** переменную **a**.

Добавим запросы **get** к нашим запросам

- **get foo a**
- **get foo c**
- **get bar a**
- **get bar b**

Продемонстрим как это могло бы выглядеть в коде

Представим как это могло бы выглядеть в коде

```
a = 0
def foo():
  b = 1
  get(a)
  get(c)
  def bar():
    a = 2
    get(a)
    get(b)
```

Результатом запроса **get** будет имя пространства, из которого будет взята нужная переменная. Например, результатом запроса **get foo a** будет **global**, потому что в пространстве **foo** не объявлена переменная **a**, но в пространстве **global**, внутри которого находится пространство **foo**, она объявлена. Аналогично, результатом запроса **get bar b** будет являться **foo**, а результатом работы **get bar a** будет являться **bar**.

Результатом **get foo c** будет являться **None**, потому что ни в пространстве **foo**, ни в его внешнем пространстве **global** не была объявлена переменная **c**.

Более формально, результатом работы **get <namespace> <var>** является

- **<namespace>**, если в пространстве **<namespace>** была объявлена переменная **<var>**
- **get <parent> <var>** – результат запроса к пространству, внутри которого было создано пространство **<namespace>**, если переменная не была объявлена
- **None**, если не существует **<parent>**, т. е. **<namespace>** – это **global**

Формат входных данных

В первой строке дано число **n** (**1 ≤ n ≤ 100**) – число запросов.
В каждой из следующих **n** строк дано по одному запросу.
Запросы выполняются в порядке, в котором они даны во входных данных.
Имена пространства имен и имена переменных представляют из себя строки длины не более **10**, состоящие из строчных латинских букв.

Формат выходных данных

Для каждого запроса **get** выведите в отдельной строке его результат.

Sample Input:

```
9
add global a
create foo global
add foo b
get foo a
get foo c
create bar foo
add bar a
get bar a
get bar b
```

Sample Output:

```
global
None
bar
foo
```

Чтобы решить это задание откройте
https://stepik.org/lesson/24460/step/10