Presentation 23

Generators

Announcements for This Lecture

Assignment 6

- These are now graded
  - **Mean**: 92, **Median**: 95
  - Within expectations
  - We removed **hard** part
- Limited regrades
  - Again major issues only
  - We are very behind here
- A7 is only thing left

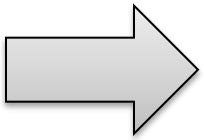
Lesson Videos

- *Almost* all are posted
  - **Lesson 28** for today
  - **Lesson 29** next time

Labs

- Today is more GUIs
- Generators will be Tues
  - Coupled with coroutines

From Last Time: More Buttons

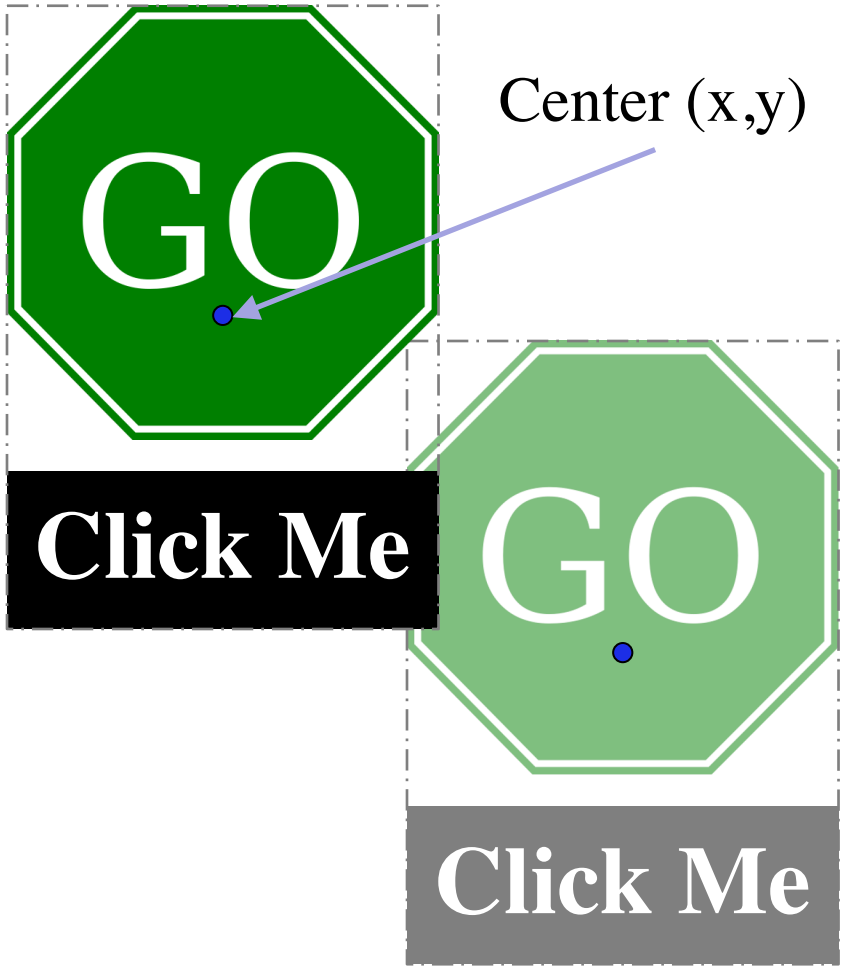


Button is **Up**

Button is **Down**

From Last Time: Composite Objects

- Way to "group" objects
  - Has a single x, y attribute
  - Moving obj moves all
- Code like subcontroller.py
  - Each object is attribute
  - Initialize them in __init__
  - Needs a custom draw
  - Update not necessary
- Used in end of **Task 1**



Activity: Call Frame Time

Function Defintions

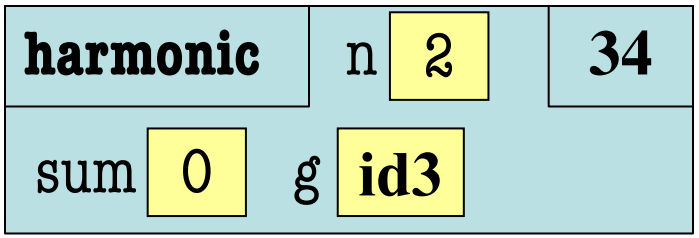
```
    def rnginv(n):        #Inverse range
19      for x in range(1,n):
20          yield 1/x

    def harmonic(n):      #Harmonic sum
32      sum = 0
33      g = rnginv(n)
34      for x in g:
35          sum = sum+x
36      return x
```

Function Call

```
>>> x = harmonic(2)
```

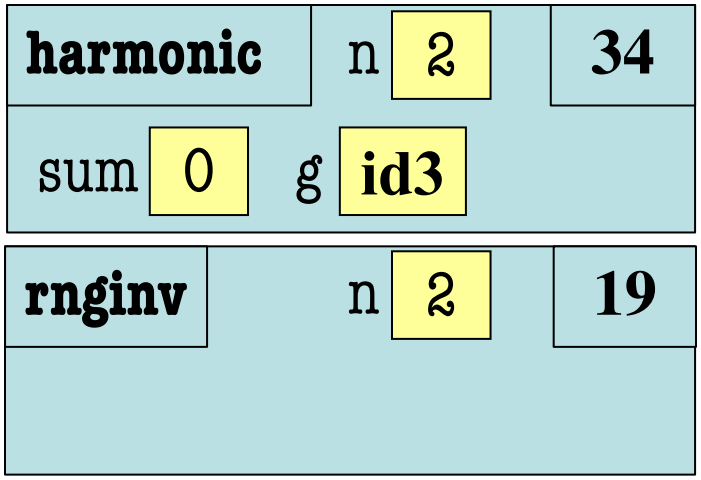
Assume we are here:



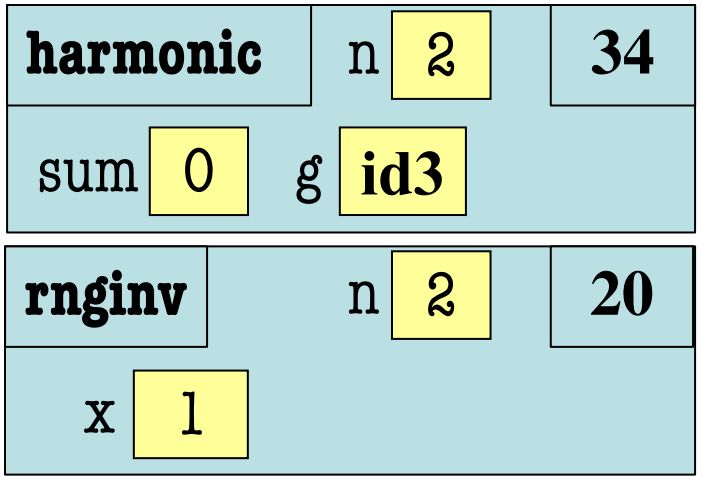
Ignoring the heap, what is the **next step**?

Which One is Closest to Your Answer?

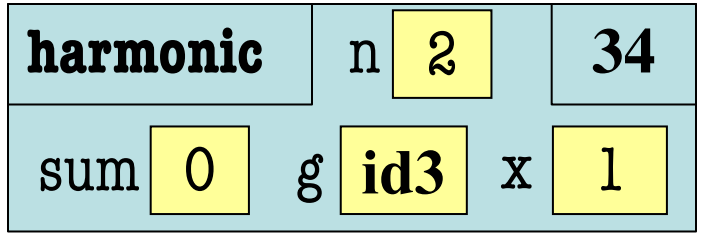
A:



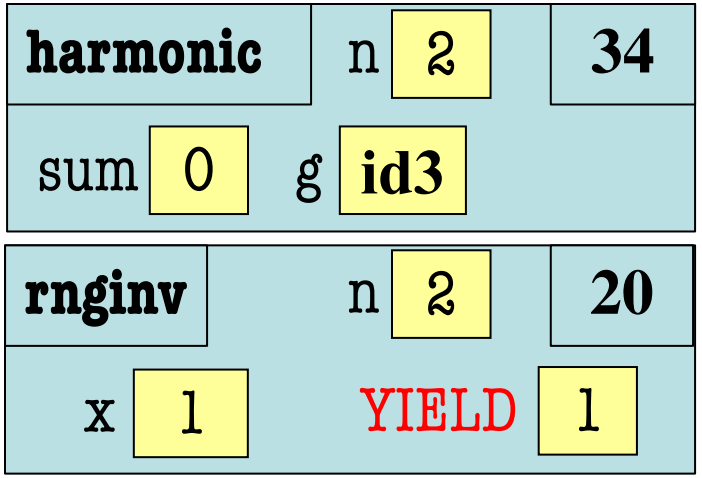
B:



C:

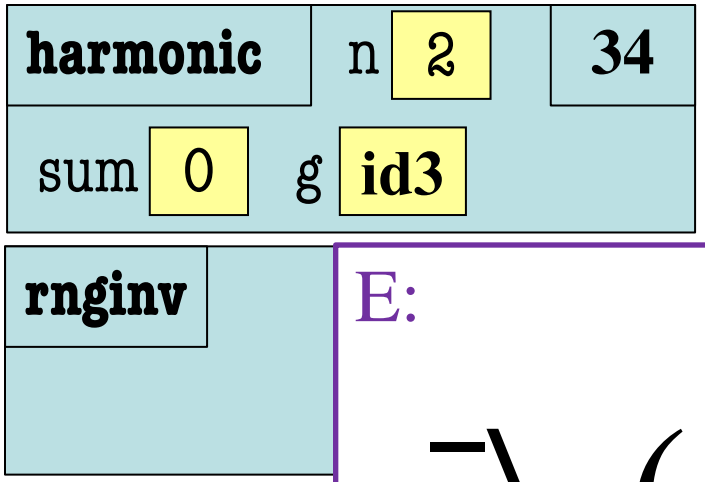


D:

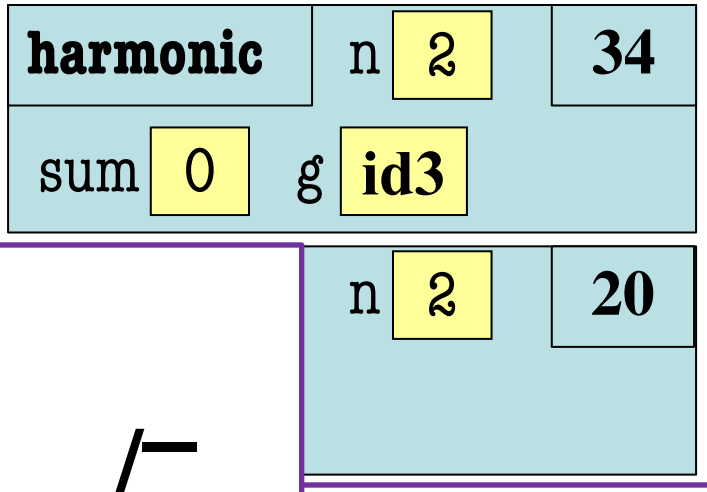


Which One is Closest to Your Answer?

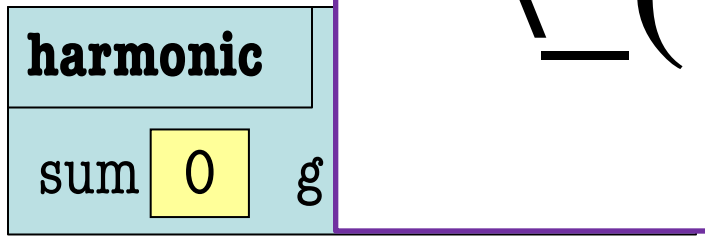
A:



B:

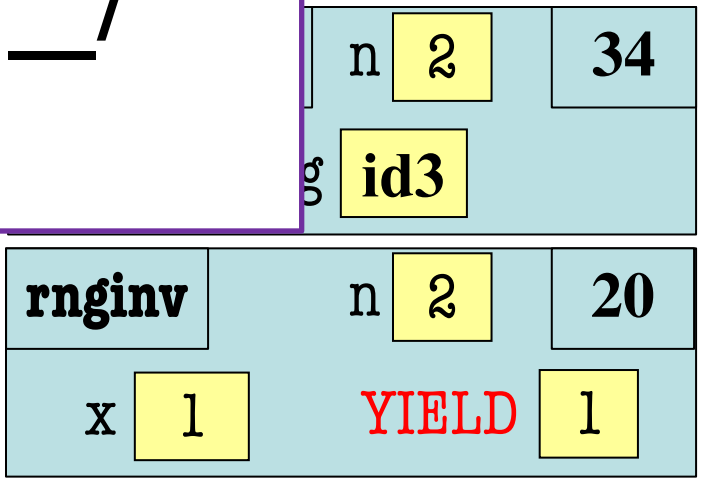


C:



E:

¯_(ツ)_/¯



Activity: Call Frame Time

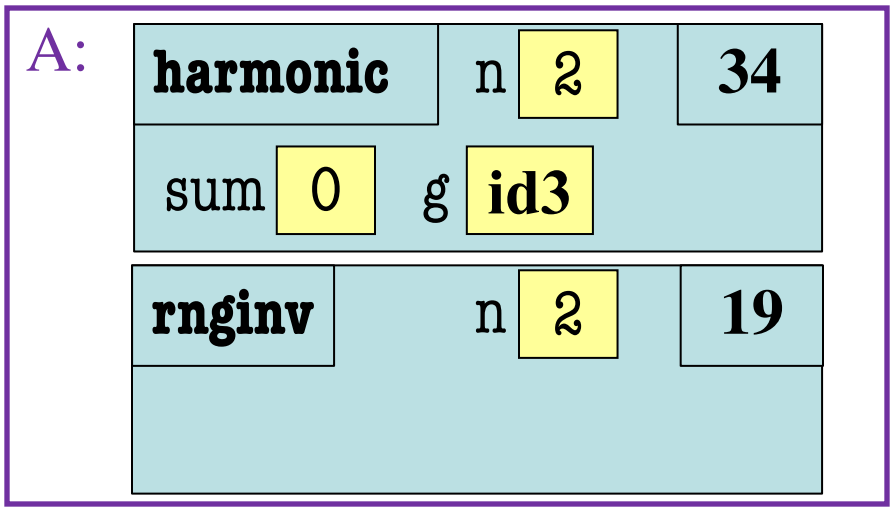
Function Defintions

```
    def rnginv(n):        #Inverse range
19      for x in range(1,n):
20          yield 1/x

    def harmonic(n):      #Harmonic sum
32      sum = 0
33      g = rnginv(n)
34      for x in g:
35          sum = sum+x
36      return x
```

Function Call

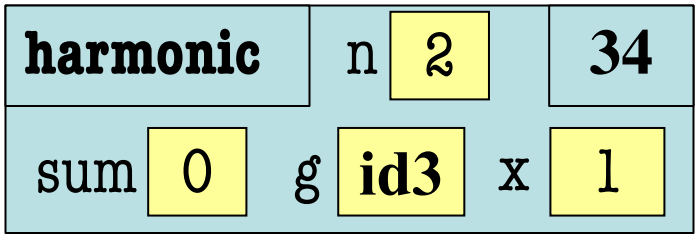
```
>>> x = harmonic(2)
```



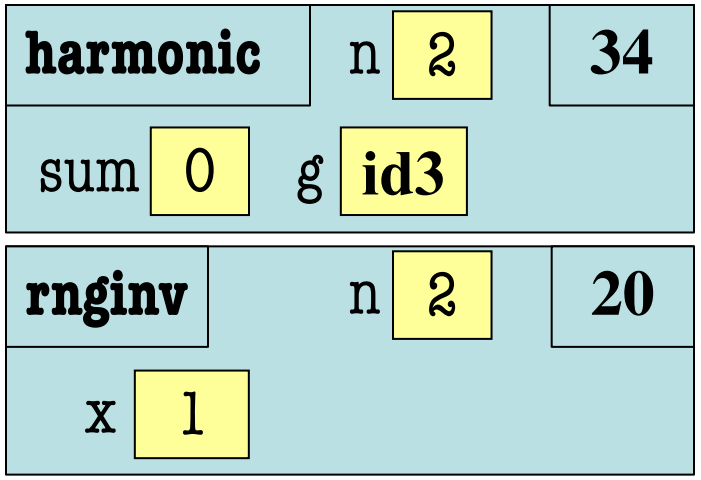
What is the **next step**?

Which One is Closest to Your Answer?

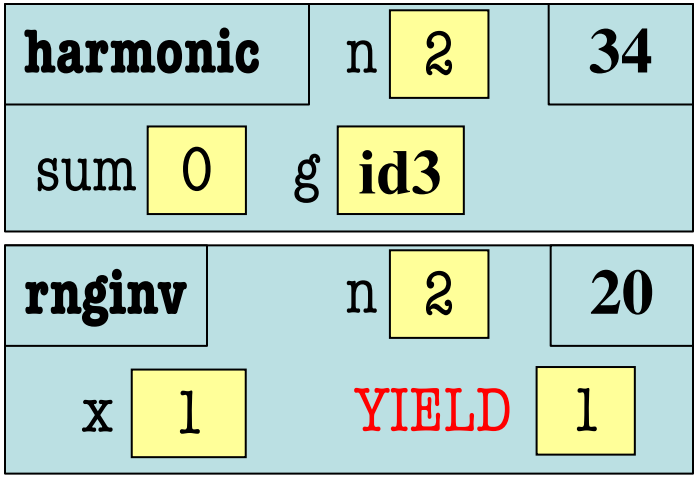
A:



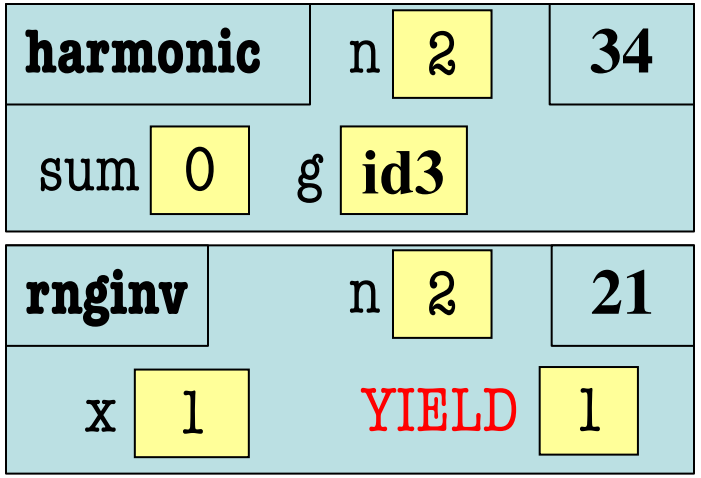
B:



C:



D:



Activity: Call Frame Time

Function Defintions

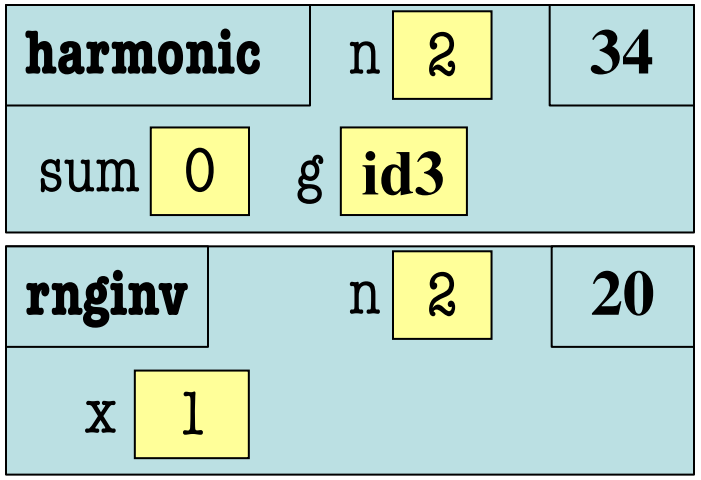
```
    def rnginv(n):        #Inverse range
19      for x in range(1,n):
20          yield 1/x

    def harmonic(n):      #Harmonic sum
32      sum = 0
33      g = rnginv(n)
34      for x in g:
35          sum = sum+x
36      return x
```

Function Call

```
>>> x = harmonic(2)
```

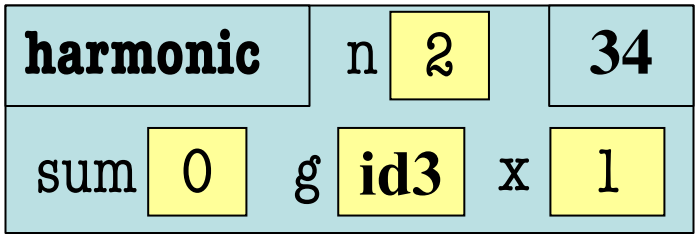
B:



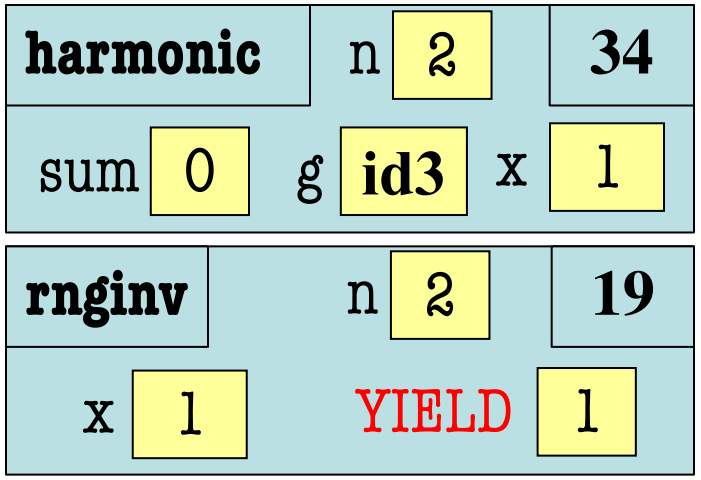
What is the **next step**?

Which One is Closest to Your Answer?

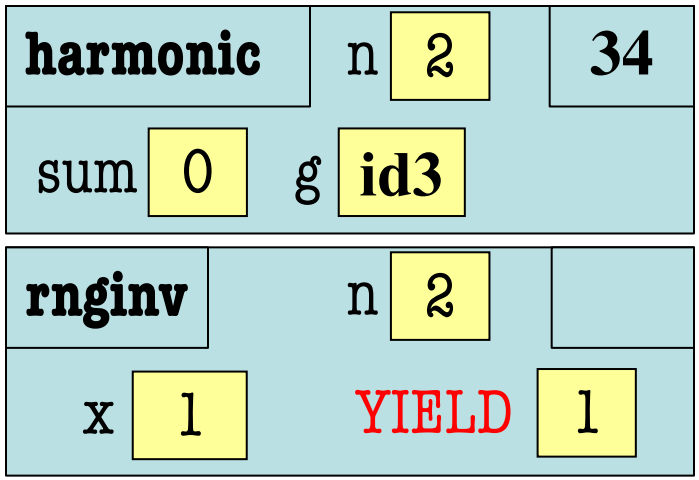
A:



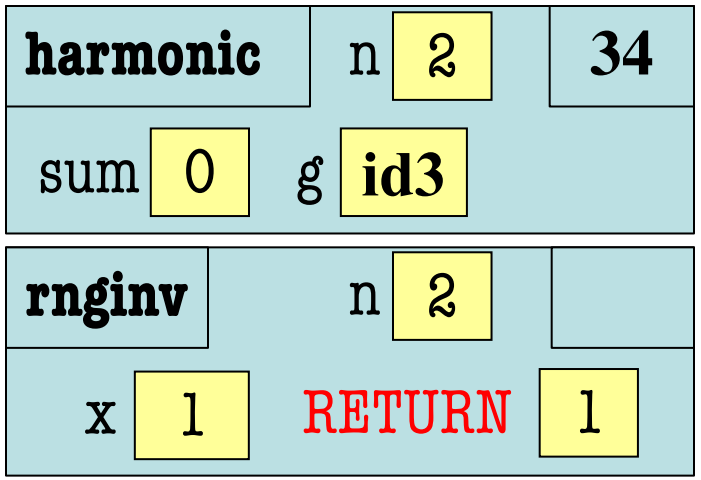
B:



C:



D:



Activity: Call Frame Time

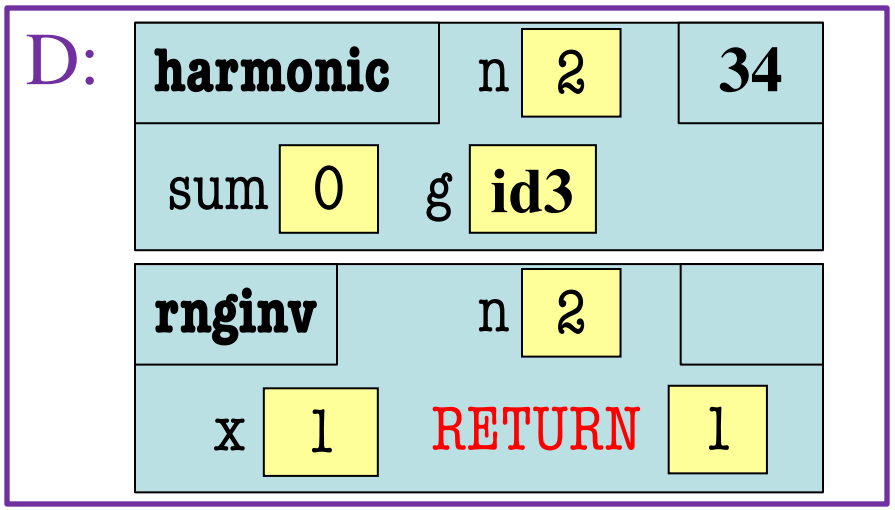
Function Defintions

```
    def rnginv(n):        #Inverse range
19      for x in range(1,n):
20          yield 1/x

    def harmonic(n):      #Harmonic sum
32      sum = 0
33      g = rnginv(n)
34      for x in g:
35          sum = sum+x
36      return x
```

Function Call

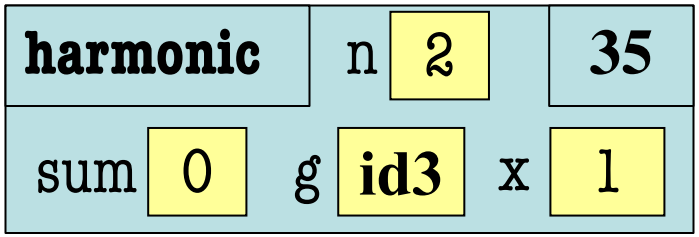
```
>>> x = harmonic(2)
```



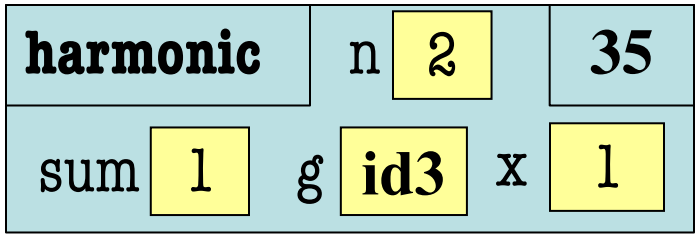
What is the **next step**?

Which One is Closest to Your Answer?

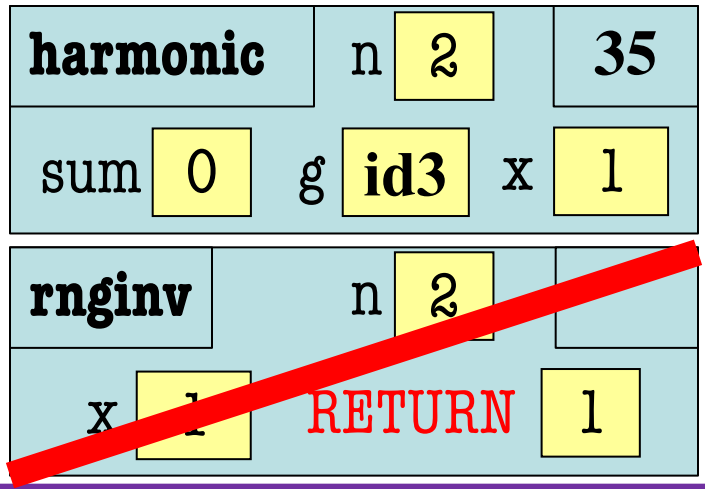
A:



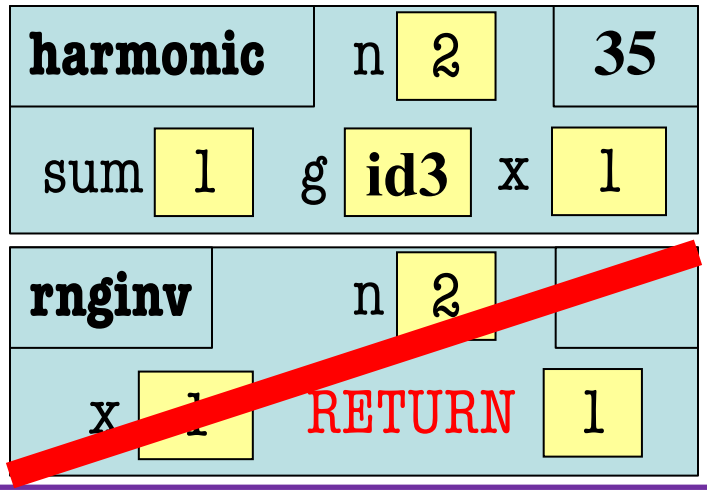
B:



C:



D:



Activity: Call Frame Time

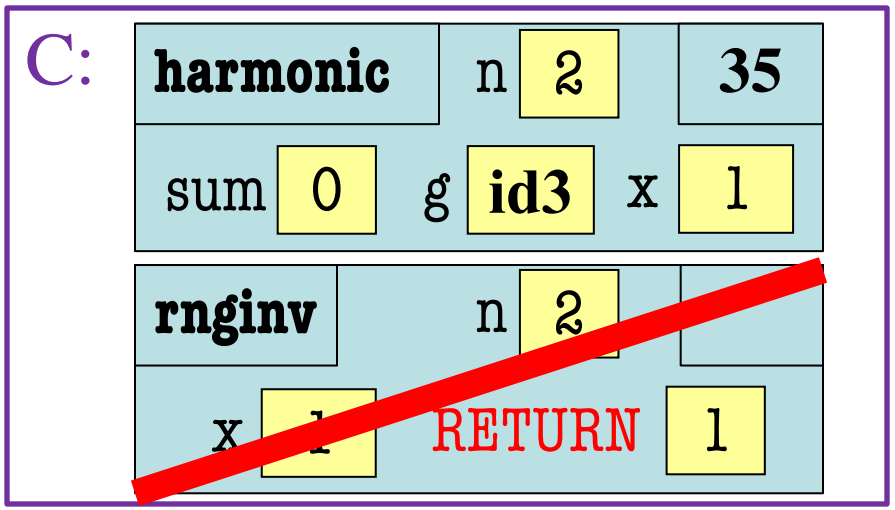
Function Defintions

```
    def rnginv(n):        #Inverse range
19      for x in range(1,n):
20          yield 1/x

    def harmonic(n):      #Harmonic sum
32      sum = 0
33      g = rnginv(n)
34      for x in g:
35          sum = sum+x
36      return x
```

Function Call

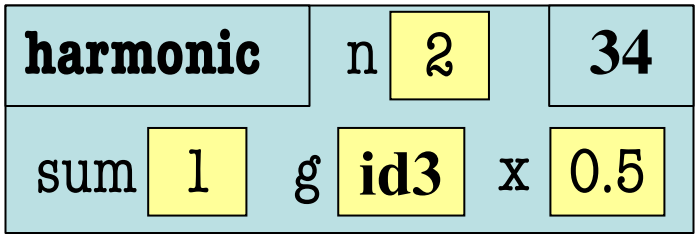
```
>>> x = harmonic(2)
```



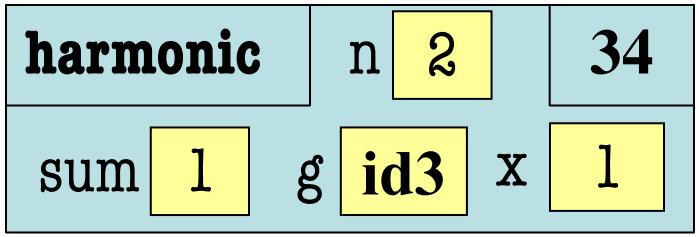
What is the **next step**?

Which One is Closest to Your Answer?

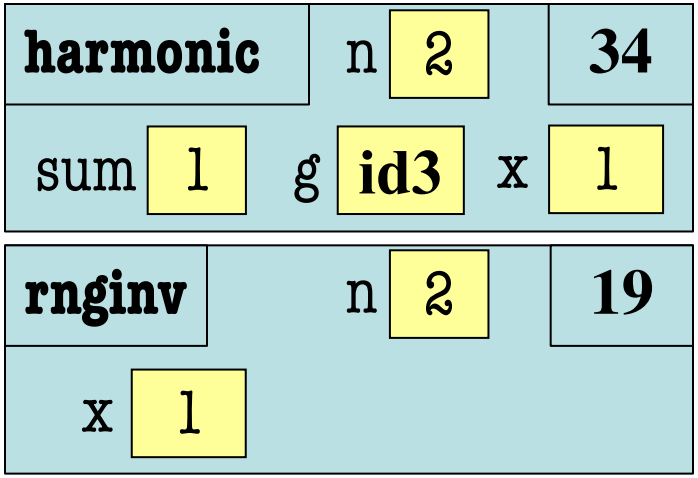
A:



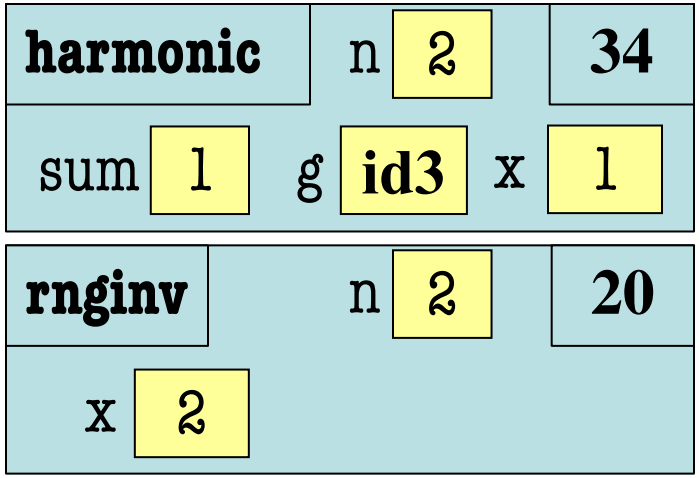
B:



C:



D:



Activity: Call Frame Time

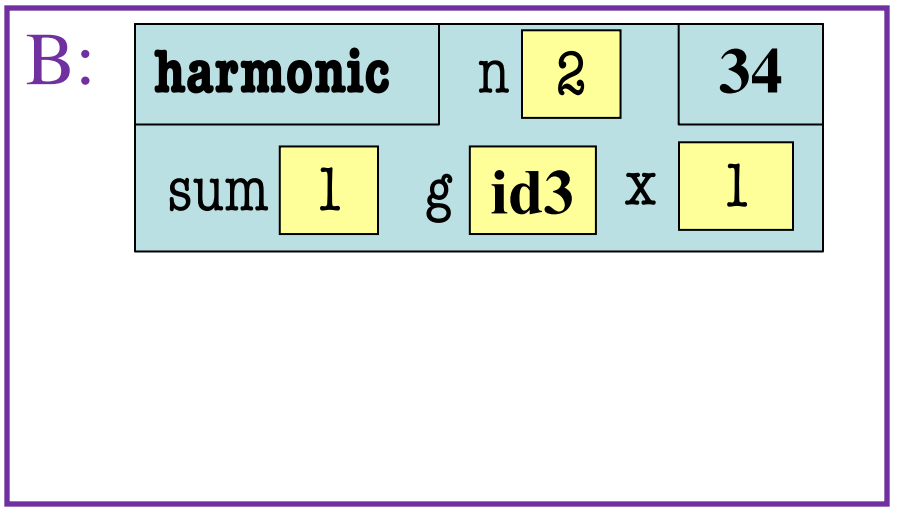
Function Defintions

```
    def rnginv(n):        #Inverse range
19      for x in range(1,n):
20          yield 1/x

    def harmonic(n):      #Harmonic sum
32      sum = 0
33      g = rnginv(n)
34      for x in g:
35          sum = sum+x
36      return x
```

Function Call

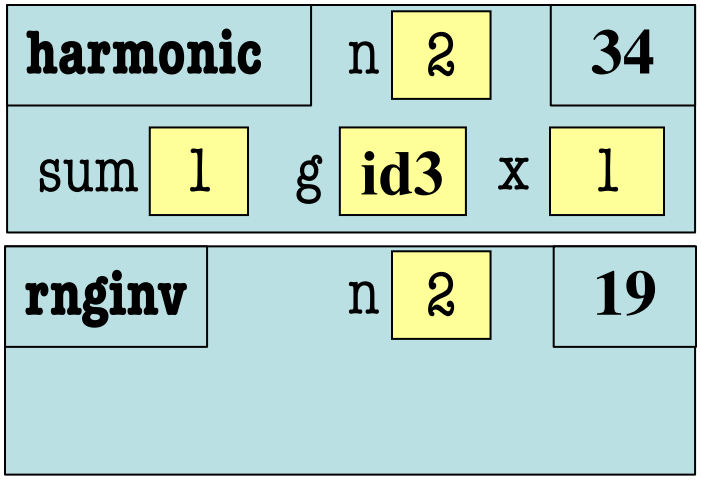
```
>>> x = harmonic(2)
```



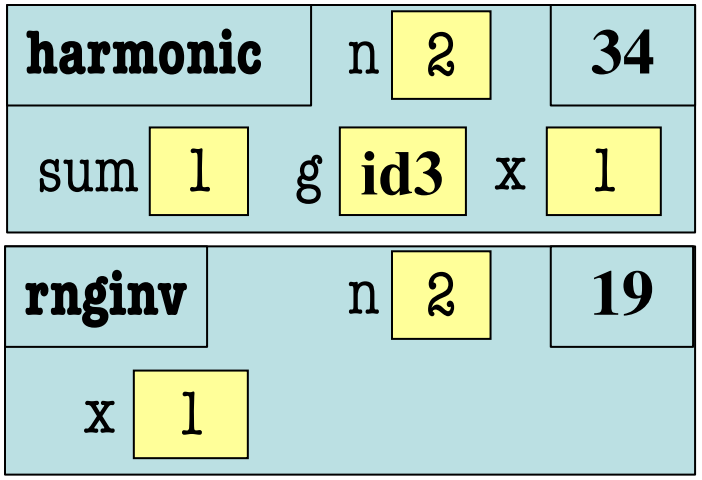
What is the **next step**?

Which One is Closest to Your Answer?

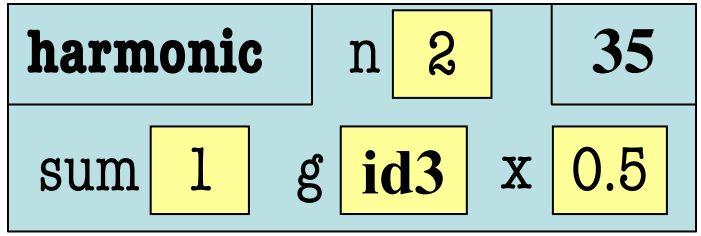
A:



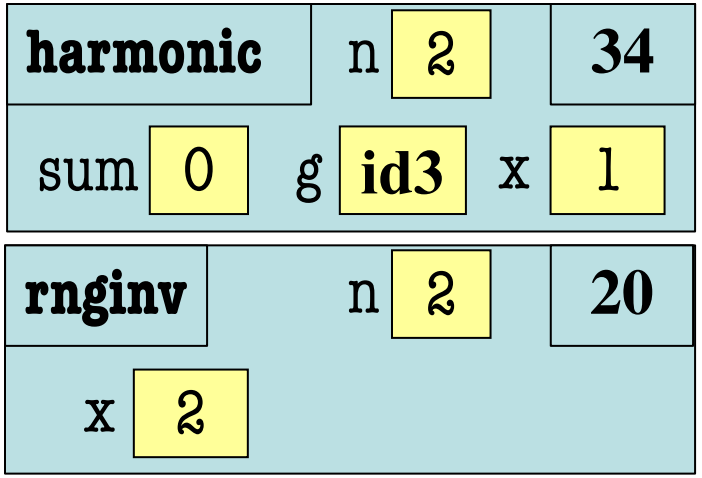
B:



C:



D:



Activity: Call Frame Time

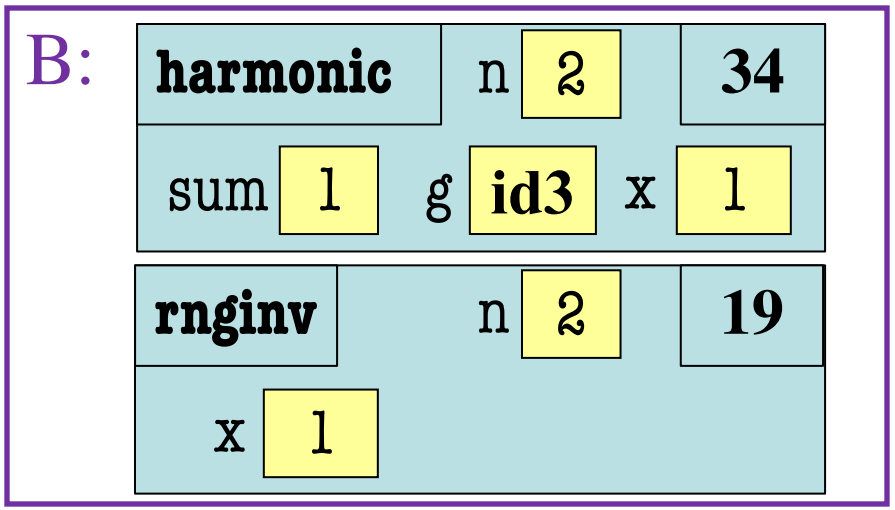
Function Defintions

```
    def rnginv(n):        #Inverse range
19      for x in range(1,n):
20          yield 1/x

    def harmonic(n):      #Harmonic sum
32      sum = 0
33      g = rnginv(n)
34      for x in g:
35          sum = sum+x
36      return x
```

Function Call

```
>>> x = harmonic(2)
```



Try the rest on your own

Functions to Generators

```
def harmonic(n):
    """
    Generates the partial sums of the harmonic series up 1/n

    The partial sum for k is 1+1/2+1/3+...+1/k

    Parameter n: The range bounds
    Precondition: n is an int > 0
    """
    pass
```

Chaining Generators

```
def sumfold(input):
    """
    Generates the sums of the numbers seen so far in input

    Example: sumfold([1,2,3]) generates the numbers 1, 3, and 6

    Parameter input: The input data to sum
    Precondition: input is a iterable of numbers (int or float)
    """
    pass
```

Chaining Generators

```
def sumfold(input):
    """
    Generates the sums of the numbers seen so far in input

    Example: sumfold([1,2,3]) generates the numbers 1, 3, and 6

    Parameter input: The input to sum
    Precondition: input is a iterable of numbers (int or float)
    """
    pass
```

For maximum **flexibility**

Chaining Generators

```
def filterdiv(input,n):
    """Generates all elements of input evenly divisble by n

    The elements are generated in the order they appear in input.

    Example: filterdiv([1,2,3,4],2) generates the numbers 2 and 4

    Parameter input: The input data to filter
    Precondition: input is a iterable of int

    Parameter n: The number to divide by
    Precondition: n is an int"""
    pass
```

Questions?