What does this code print?

```python
def a(n):
    return 2 * n

def b(n):
    return 3 * n

def c(x):
    return a(a(x))

def d(x):
    return c(x) + b(x)

print(d(a(b(1))))
```

(Your response)
def double(n):
    return 2 * n

def triple(n):
    return 3 * n

def quadruple(x):
    return double(double(double(x)))

def septuple(x):
    return quadruple(x) + triple(x)

print(septuple(double(triple(1))))
def double(n):
    return 2 * n

def triple(n):
    return 3 * n

def quadruple(x):
    return double(double(x))

def septuple(x):
    return quadruple(x) + triple(x)

print(septuple(double(triple(1))))

import numbers
numbers.double(2)

Modules are namespaces
Why object-oriented programming?
OOP terminology
Object-oriented programming languages differ in:

- how the programmer specifies an object's **interface**
- how the programmer specifies an object's **implementation**
- how objects are created, **initialized, queried, and updated**
- **encapsulation** mechanism
  how strictly the language enforces the separation between interface & implementation
Object-oriented programming (in Python)
Object-oriented programming languages differ in:

- how the programmer specifies an object's **interface**
- how the programmer specifies an object's **implementation**
- how objects are **created**, **queried**, and **updated**
- **encapsulation** mechanism
  - how strictly the language *enforces* the separation between interface & implementation
It’s all namespaces!
Interface for a course

- Each course has:
  - a number (e.g., 42)
  - a name (e.g., Principles and Practices of Computer Science)

- We can:
  - create a course (initializing it with its number and name)
  - access / change a course’s number
  - access / change a course’s name
  - determine if a course is intro-level
    true if the course’s number is ≤ 100
  - print a course
class Course:
    '''Represents a course at Ivy Tech State (go Platypuses!)'''

    HIGHEST_INTRO_LEVEL = 100

    def __init__(self, number, name):
        self.number = number
        self.name = name

    def isIntro(self):
        '''Returns True if this is an introductory-level course'''
        return self.number <= Course.HIGHEST_INTRO_LEVEL

    def __str__(self):
        return '{}: {}'.format(self.number, self.name)
Creating, accessing, and modifying a course

create an instance by “calling” the class (calls __init__)
1 cs42 = Course(42, 'Principles and Practices of Computer Science')

2 print(cs42.number)     # attribute access

3 print(cs42.isIntro())   # method call

4 cs42.number = 1000      # object modification

5 print(cs42)             # calls __str__
Interface for a student

- Each student has:
  - a name (e.g., Ben)
  - a number (e.g., 101010101)
  - a collection of courses that the student has registered for

- We can:
  - create / initialize a student instance
  - access / change a student’s name
  - access / change a student’s number
  - access a list of a student’s courses
  - register the student for a course
    - only if the student is registered for < 5 courses
  - drop a student from a course