CS 120 (Fall 21): Introduction to Computer Programming II

Short Project #4 - Intro to References

due at 5pm, Thu 16 Sep 2021

1 Overview

In this project, you will be practicing with reading and creating reference diagrams. First, we will give you a series of reference diagrams, and you will produce a function to build each one; some of the functions will have parameters, while others will not.

Second, we will give you a few snippets of code, and **you** will draw the reference diagram for each; in some cases, you will simply produce a reference diagram for the final state, but in others you will produce multiple diagrams, to represent various steps in the process.

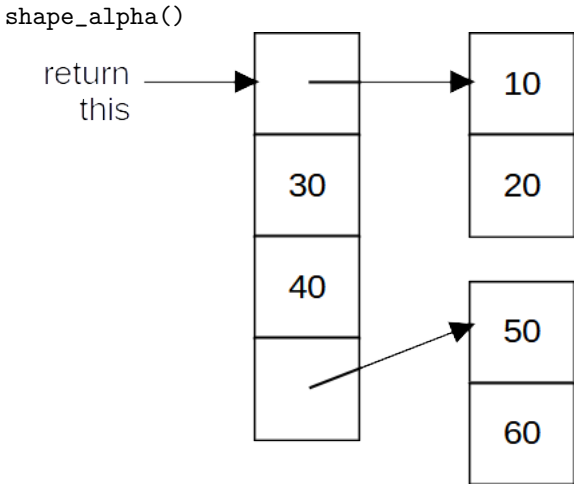
2 Part 1: Shape Functions

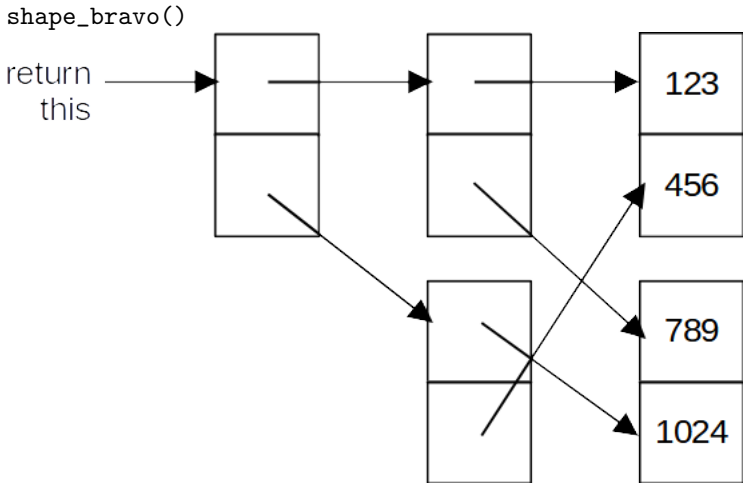
In `shapes.py` write a series of functions, each of which builds a data structure of some sort; we've drawn the picture for each data structure below. Some of the functions take one or more parameters; in that case, drop the values into the data structure at the proper location. You should **not** do anything to these parameters (other than putting them in the proper locations); you know nothing about what they contain.

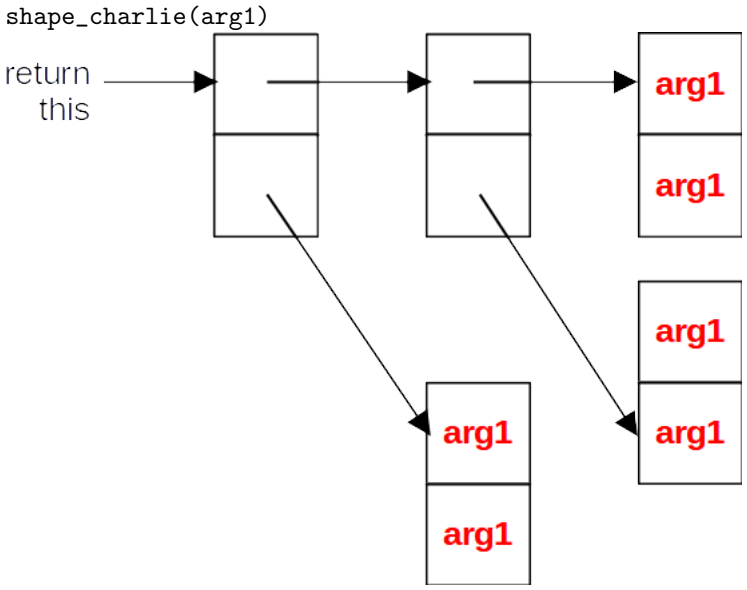
NOTE: All of the rectangles in these pictures represent Python lists (that is, arrays). Do not attempt to use tuples, classes, or any other type.

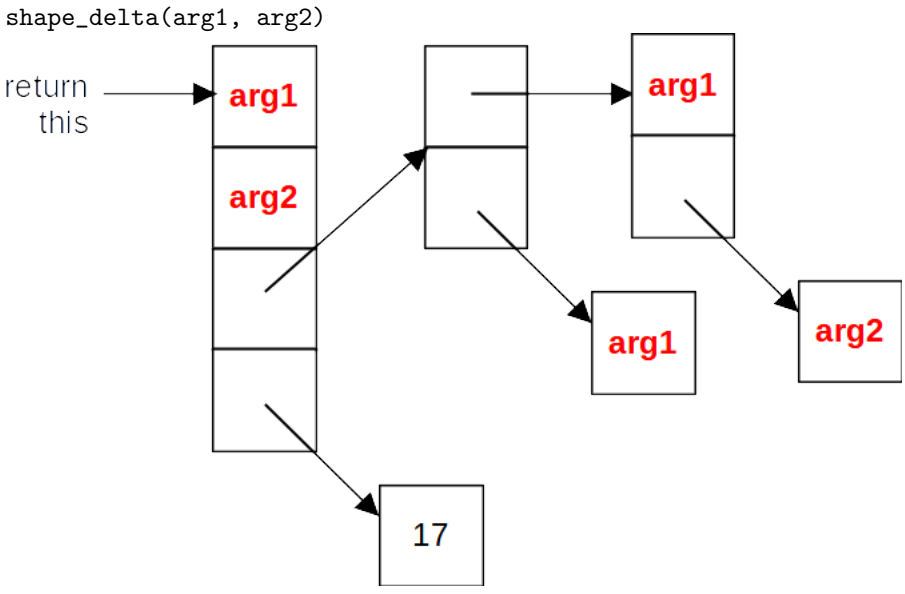
NOTE: Notice that some of these functions take parameters. Those that take parameters will put those parameters at specific locations within the shape that you must generate.

(diagrams begin on the next page)

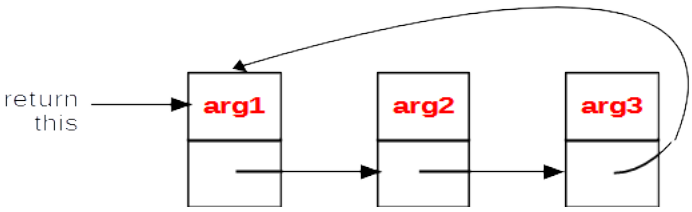

shape_alpha()
return this
10
20
30
40
50
60


shape_bravo()
return this
123
456
789
1024


shape_charlie(arg1)
return this
arg1
arg1
arg1
arg1
arg1
arg1


shape_delta(arg1, arg2)
return this
arg1
arg2
arg1
arg1
arg2
17

```
shape_echo(arg1, arg2, arg3)
```



(spec continues on next page)

3 Part 2: Draw Your Own Reference Diagrams

For these diagrams, you may either submit a set of image files, or a single PDF that has all of them. Whatever you do, make sure to clearly label each one.

3.1 Diagram A

Draw a reference diagram for the **final state** of the variables, after this snippet of code runs.

```
data = [None, None, None]

for i in range(2):
    data[i] = [3*i, 2*i, 1*i]

data[2] = data
```

3.2 Diagram B

Draw a reference diagram for the final state of the variables, after this snippet of code runs.

```
data       = [10, None, 20, ["foo", None, "baz"], None ]
data[1]    = "asdf"
data[4]    = data[3]
data[3][1] = [1,2,3]
```

3.3 Diagram(s) C

Draw multiple reference diagrams for this snippet of code; draw one for each step where you see the `DRAW HERE` comment. (Inside the loop, draw it once for each iteration of the loop.)

For this problem, **also write out what this code will print.**

```
all  = [10, None]
tail = all
# DRAW HERE

for i in range(4):
    tail[1] = [ i*10+20, None ]
    tail    = tail[1]
    # DRAW HERE

print(all)
```

(spec continues on next page)

4 Turning in Your Solution

You must turn in your code using GradeScope.

5 Acknowledgements

Thanks to Saumya Debray for many resources that I used and adapted for this class.