SI 413 Fall 2021: Homework 10

Due Monday, November 15

Your name:

Citations and collaborators:

Comments, suggestions, or questions for your instructor:

Fill out the first row of the table on a 0-5 scale before turning in.

This rubric is also available on the website under "Admin":

- **5**: Solution is completely correct, concisely presented, and neatly written.
- **4**: The solution is mostly correct, but one or two minor details were missed, or the presentation could be better.
- **3**: The main idea is correct, but there are some significant mistakes. The presentation is somewhat sloppy or confused.
- **2**: A complete effort was made, but the result is mostly incorrect.
- **1**: The beginning of an attempt was made, but the work is clearly incomplete.
- **0**: Not submitted.

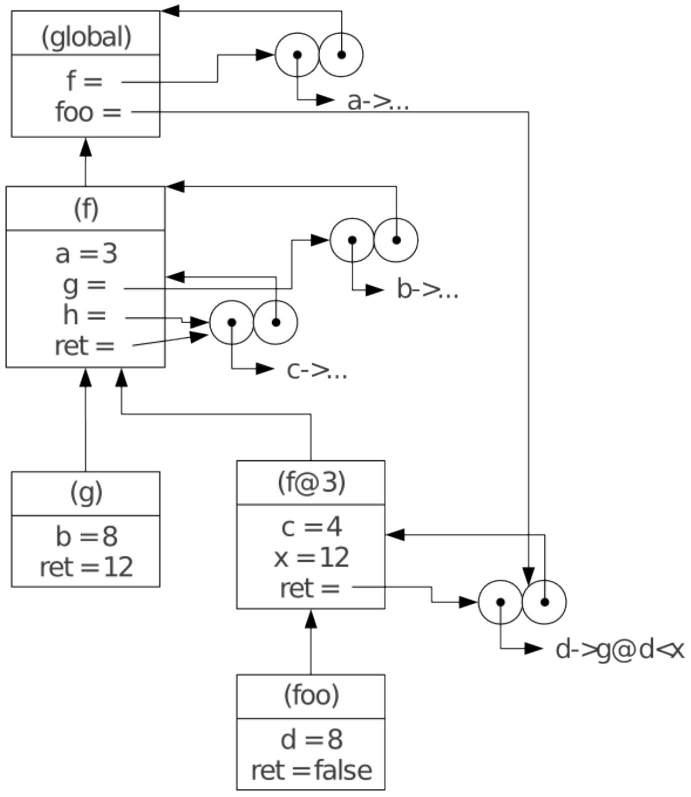
Problem	1	2	3	Total
Self-assessment				▨
Final assessment				

1 Reference Count Practice

Consider the following SPL code:

```
new f := lambda a {
  new g := lambda b { ret := b + b/2; };
  new h := lambda c {
    new x := a*c;
    ret := lambda d { ret := g@d < x; };
  };
  ret := h;
};
new foo := f@3@4;
write foo@8;
foo := 20;
```

Here are the frames and closures that exist *just before the last line is executed.* (Note: it would be good practice to see if you could recreate this diagram yourself!)



(Questions are on the next page.)

a) Using the labels of each frame above, indicate what the reference count for each frame is at this point in the program.

b) Repeat (a), showing what happens to the reference counts after the last line in the program is executed.

2 Mark & Sweep Practice

For this problem, use the same example program and frames-and-closures diagram as for the first problem.

a) Using the labels of each frame above, indicate which frames would be garbage collected at this point using the *mark and sweep* method.

b) Repeat (a), showing what happens in mark and sweep after the last line in the program is executed.

3 Garbage 101

Write a short program in any programming language *other than* SPL that creates exactly 101 objects of garbage which cannot be collected using normal reference counting.

That is, your program should somehow cause at least 101 objects to be allocated on the heap, which are unreachable at the end of the program and therefore considered “garbage”, but which will not be collected by a naïve reference counting garbage collector implementation.

Clearly indicate the programming language you choose and try to keep it simple!