Due: Wednesday, November 20

Instructions: Review the course honor policy for written homeworks.

This cover sheet must be the front page of what you hand in. Fill out the left column in the table to the right after we go over each problem in class, according to the rubric below.

This rubric is also on the website, in more detail, under “Other Stuff”→“Grading Rubrics”.

Make sure all problems are submitted IN ORDER.

• 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. There may be some basic misunderstandings of the topic or the problem.
• 1: The beginning of an attempt was made, but the work is clearly incomplete.
• 0: Not submitted.

Comments or suggestions for the instructor:

What other students did you work with?

Citations (be specific about websites):
Consider the following SPL code:

```java
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.

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

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

2 Cooking Spaghetti

In class we talked about “spaghetti code”, which is code that uses GOTO statements way too liberally and becomes almost impossible to follow. I want you to write your own example of spaghetti code, in the C++ language.

Submit your code electronically as 413 hw 10 in a file called ex2.cpp. Your program should compile and run without any arguments or special flags. You also need to submit a description of what your code does, written out and turned in with the other exercises for this assignment.

Grading this problem will consist of me first looking at your code, briefly, and trying to figure out what it does. I will then look at your description of what it actually does and then run it. If your code actually does something, and it’s what you describe, but I couldn’t discern that by looking at it, then you’ll do well on this problem.

It may help to go back and read Dijkstra’s original article on GOTOs.