

## Written Assignment 3

**Assigned:** February 24**Due:** March 1 at 11:59pm

**Instructions:** This assignment asks you to prepare written answers to questions on semantic analysis. Each of the questions has a short answer. You may discuss this assignment with other students and work on the problems together. However, your write-up should be your own individual work.

Please write your name, email address, and discussion section on your homework. ***Please start each question on a new page. All written assignments must be submitted as a PDF via Gradescope: <https://gradescope.com>***. Instructions for how to submit assignments to Gradescope can be found at the following links: [https://gradescope.com/get\\_started#student-submission](https://gradescope.com/get_started#student-submission)

1. Examine the following ChocoPy program and list all erroneous lines. For each erroneous line, briefly explain the error. (Note: For this question, only include compile-time errors caught during semantic analysis.):

```
1. x:int = 1
2. y:int = 2
3. z:Dog = None
4. class Dog(object):
5.     y:str = "y"
6.     def f(x:int)->int:
7.         y:int = 3
8.         nonlocal x
9.         x = 0
10.        return x
11.     def g(y:int):
12.         global y
13.         y = 3
14.     g(y)
15. f(len(z.y))
```

2. The Java language has arrays and allows inheritance. In Java you are allowed to assign a value of type  $T[]$  to a variable of type  $U[]$  so long as the type  $T$  is a subtype of  $U$ . This however, turns out to be unsound despite being able to pass Java's type checking rules. Give an example of Java code that would pass Java's static type checking but would cause an error in execution. Also describe why this error occurs in your example.

3. Write Java code that would print “dynamic” if executed with dynamically scoped environments, but “static” if executed with statically scoped environments.

4. Imagine that we have a series of user defined types in ChocoPy as follows: **Animal** is a class, and **Dog**, **Cat**, and **Bear** all derive from the **Animal** class. There are two classes **Poodle** and **Pug** which derive from the **Dog** class. There are two classes **Siamese** and **Persian** which derive from the **Cat** class. There is one class **Oski** which derives from the **Bear** class.
- (a) Let **po** be of static type **Poodle** and let **pu** be of static type **Pug**. What is the type of **[po, pu]** (a list containing **po** and **pu**)?
  - (b) Let **o** be of static type **Oski** and let **b** be of static type **Bear**. What is the type of **[o, b]**?
  - (c) Let **p** be of static type **Pug** and let **s** be of static type **Siamese**. Also let **o** be of static type **Oski**. What is the type of **[p, s, o]**?

5. The following ChocoPy program satisfies all type-checking rules. Express the typing derivation of this program in an inverted-tree representation:

```
a : int = 0
b : int = 0
for a in [1, 2]:
    b = a + b
```

That is, the the judgment that the program is well-typed should be at the bottom of the tree, and any hypotheses (which are themselves typing judgments) used for this judgment should be just above, and so on. You can omit  $M, C$ , and  $R$  in the typing judgments. Assume the initial environment is  $O = \{int/a, int/b\}$ .

6. In Question 1, you were asked to find the erroneous lines in the program, which largely dealt with issues of scope. Now we will focus on typing rules. For the following ChocoPy program, please list the lines (if any) that do not type check, and provide brief justification.

```
1. x : int = 3
2. y : str = "a"
3. def f() -> object:
4.     return 0
5. y = y + x
6. x = None
```