Type Checking Exercises
CS 364 — Spring 2022

1 Definitions and Background

1. Define the following terms and give examples where appropriate.

   (a) Semantic Analysis:

   (b) Variable Declaration:

   (c) Variable Definition:

   (d) Scoping Rules:

   (e) Symbol Table:

   (f) Type:

   (g) Type System:

   (h) Type Checking:

   (i) Type Inference:

   (j) Soundness:

(k) Dynamic Type:

(l) Static Type:

2. Describe some key differences between statically- and dynamically-typed languages. What are some advantages of each approach?

2 Scopes

1. Draw the AST for the following snail snippet and annotate the variables that are in scope. You may assume that class E has been defined an that it has a function `set_var`, which has a single formal parameter.

```
{
  let num = in_int();
  let x = 1;

  {
    let y = 1;
    while (y <= num) {
      x = x * y;
      y = y + 1;
    };
  };

  (new E).set_var(x);
}
```

2. Give an example program that will produce different output depending on whether the programming language implements static or dynamic scoping. What is the output in each case?

3 Type Checking

1. The Java programming language includes arrays. The Java language specification states that if s is an array of elements of class S, and t is an array of elements of class T, then the assignment $s = t$ is allowed as long as T is a subclass of S.

   This typing rule for array assignments turns out to be unsound. Java works around this by inserting runtime checks to throw an exception if arrays are used unsafely.

   Consider the following Java program, which type checks according to the preceeding rule:

```
1  class Mammal { String name; }
2
3  class Dog extends Mammal {
4    void beginBarking() { ... }
5  }
6
7  class Main {
8    public static void main(String argv[]) {
9      Dog x[] = new Dog[5];
10     Mammal y[] = x;
11
12     // Insert your code here
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40   }
41 }
```

   Add code to the `main` method so that the resulting program is a valid Java program (i.e., it compiles), but running that program triggers one of the aforementioned runtime checks. Include a brief explanation of how your program exhibits the problem.