EECS483 — Review Set 3

These review questions are meant to familiarize you with semantic analysis. These are representative of questions that may be asked on the second midterm. Please come by office hours if you have questions.

1. C++, unlike Cool, supports multiple inheritance. For example, the following class hierarchy is legal in C++:

$$\begin{tikzcd} & A \arrow[dl] \arrow[dr] & \\ B \arrow[dr] & & C \arrow[dl] \\ & D & \end{tikzcd}$$

   Allowing multiple inheritance changes the way we define and use the least upper bound (lub) function on types.

   (a) Explain, using at least one example, why it is necessary to change *lub*.

   (b) Describe (briefly) how you might implement *lub* for multiple inheritance.

2. The Java programming language includes arrays. The Java language specification states that if s is an array 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 preceding rule:

```java
class Mammal {String name; }
class Dog extends Mammal {
    void beginBarking() { ... }
}

class Main {
    public static void main (String argv[]) {
        Dog x[] = new Dog[5];
```

```java
        Mammal y[] = x;
        //Insert your code here
    }
}
```

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 of the aforementioned runtime checks. Include a brief explanation of how your program exhibits the problem.

3. The following typing judgments have one or more flaws. For each judgment, list the flaws and explain how they affect the judgment with respect to the Cool language.

   (a)

$$\frac{\begin{array}{l} O \vdash e_0 : T \\ O \vdash T \leq T_0 \\ O \vdash e_1 : T_1 \end{array}}{O[x/T_0] \vdash \text{let} x : T_0 \leftarrow e_0 \text{ in } e_1 : T_1} \; [let - init]$$

   (b)

$$\frac{\begin{array}{l} O(\text{id}) = T_0 \\ O \vdash e_1 : T_1 \\ T_0 \leq T_1 \end{array}}{O \vdash \text{id} \leftarrow e_1 : T_1} \; [assign]$$

   (c)

$$\frac{T \leq C}{\vdash \texttt{SELF\_TYPE}_C \leq T} \; [self - type]$$

   (d)

$$\frac{\begin{array}{l} O, M, C \vdash e_0 : T_0 \\ ... \\ O, M, C \vdash e_n : T_n \\ T_0 \leq T \\ M(T_0, f) = (T'_1, ...T'_n, T'_{n+1}) \\ T'_{n+1} \neq \texttt{SELF\_TYPE} \\ \forall 1 \leq i \leq n \; . \; T_i \leq T'_i \end{array}}{O, M, C \vdash e_0 @ T.f(e_1, ..., e_n) : T'_{n+1}} \; [static - dispatch]$$