1. [9] A miniJava program and its corresponding AST are shown on the reverse side of this page.

   a) On the right hand side of the AST, number each node of the AST (i.e. line of text in the display format) that should have a Declaration attribute. The numbering should follow the order in which the nodes are assigned a controlling declaration during identification. Draw an arrow (again on the right hand side) from each numbered statement to its controlling declaration.

   b) On the left hand side of the AST, number each node in the AST that should have a TypeDenoter attribute and write its type. The numbering should follow the order in which the TypeDenoter attributes are assigned.

2. [6] Suppose we have the following type declarations for variables and functions

```
+ : Int × Int → Int
* : Int × Int → Int
+ : Real × Real → Real
* : Real × Real → Real
== : α × α → Bool
== : α × α → Stmt
r : Real
i : Int
```

Assume also that integer or real literals have a single (manifest) type (e.g. “2” is Int and “2.0” is Real). Use the type checking framework for simple ASTs described in Lecture 11 for the following.

(a) Draw the AST and perform type assignment for the statement \( r = r \times 2 \)

(b) Draw the AST and perform type assignment for the expression \( i == 2 \times i \)
```java
class Test {
    public int go() {
        Foo p = new Foo();
        this.p = p;
        return p.x;
    }
    public Foo p;
}

class Foo {
    public int x;
}
```

Again, AST Display

```java
class Test {
    public int go() {
        Foo p = new Foo();
        this.p = p;
        return p.x;
    }
    public Foo p;
}
```