CS502: Compiler Design

Fall 2022 (Due: September 24th, 2022)

Assignment A2: You're not my type

---

1 Assignment Objective

Implement Type-Checking and list all the type related errors for a given program.

2 It's a Sunday, I don't move on Sundays

Joey was able to complete the parser implementation with your help in the last assignment. Now his next task is to find type checking errors in the code but he keeps asking for Chandler's help. Chandler is fed up of manually finding these errors, so he decides to incorporate a Type-Checking stage into the compiler. Help Chandler build this Type-Checking system by using this specification.

3 Detailed Specification

You are provided with a grammar file `Tchk.jj`, which is a similar to the last assignments grammar with some simplifications (arrays do not exist in this language). Your job is to incorporate the following Type checks in the given grammar.

- **Assignment Statement:** Make sure that the *Type* of the right hand side is compatible with that of the left hand side.

```
class Table {...}
class Animal {...}
class Monkey extends Animal {...}
...
int a;
Animal o;
o = new Monkey(); // No error, as Monkey is a subtype of Animal
o = new Table();  // Type checking error, assigning Table to Monkey object
a = true;         // Type checking error, assigning bool to integer
```

- **Invalid Binary Operations:** Make sure the binary operations are valid.

```
int a;
Animal o;
a = 10 + 55;      // No error
a = 10 + o;       // Type checking error, addition of int and Animal object
```

- **Control Flow Statements:** Make sure the condition check is a boolean expression.

```
int a;
int b;
if (a > b) {...}  // No error
if (a + b) {...}  // Type checking error, check should be a boolean
```

- **Function calls:** Perform argument matching at function call sites to make sure correct type of arguments are supplied. Also make sure the function return type is valid.

```
int foo(Animal a, bool b) {... return 10;} // check the return type aswell
...
Monkey m;
Table t;
bool res;
int r;
r = foo(m, true);  // No error
r = foo(t, true);  // Type checking error, Table is not a subtype of animal
res = foo(m, true);// Type checking error, assigning int to bool
```

The following information is to be printed after identifying all the type errors.

- Assignment: No. of Assignment statement type check errors
- Binop: No. of Binop errors
- Control: No. of control flow related errors
- Function: No. of function call related errors

3.1 Example Input

```
class Test1 {
    public static void main(String[] args) {
        System.out.println(10+true); // 1 Binop error
    }
}

class A {
    public int bar(boolean b) {
        int a;
        float f;
        a = 10;
        f = 10;
        // NOTE: Assignment of int to float is allowed, but not vice versa
        f = a;
        a = f; // 1 Assignment error
```

```
            a = b; // 1 Assignment error
            if (a + f) {} // 1 Control error
            return b; // 1 Function error
        }

        public int test(A o1) {
            return 0;
        }

        public boolean foo(int p) {
            int a;
            boolean c;
            Test1 t;
            A o1;
            B o2;
            C o3;
            c = false;
            t = new Test1();
            o1 = new B();
            o2 = new B();
            o3 = new C();
            a = ((10 + (10.1 + t)) + 10); // 3 Binop errors, 1 Assignment error
            p = 20;
            p = this.bar(c);
            p = this.test(o1);
            p = this.test(o2);
            p = this.test(o3); // 1 Function error, 1 Assignment error
                               // Each argument matching failure
                               // will be a separate Function error
            c = a + p; // 1 Assignment error
            return c;
        }
    }

    class B extends A {}

    class C {}
```

3.2 Example Output

```
Assignment: 5
Binop: 4
Control: 1
Function: 2
```

3.3 Notes

There are some general notes and assumptions that you can make regarding the test cases.

- There will be no function overloading/overriding.
- There will be no invalid function calls i.e. calls to undeclared functions.
- Assignment of int to float is allowed in Java but not vice versa.
- We have provided a TypeCheckVisitor class that can be used for this assignment (not mandatory).

3.4 Evaluation

Your submission must be named as `rollnum-a2.zip`, where `rollnum` is your roll-number in small letters. Upon unzipping the submission, we should get a directory named `rollnum-a2`. The main class inside this directory should be named `Main.java`. Your program should read from the standard input and print to the standard output. You can leave all the visitors and syntax-tree nodes as it is, but remember to remove all the `.class` files and jar files.

We would run the following commands as part of the automated evaluation process:

- `javac Main.java`
- `java Main < test > out`

If the contents of `out` match with the expected output for the testcase, you would get marks for the corresponding testcase.

4 Plagiarism Warning

You are allowed to discuss publicly on class, but are supposed to do the assignment completely individually. We would be using sophisticated plagiarism checkers, and if similarity is found, the penalty used in the course would be as follows:

- First instance: 0 marks in the assignment
- Second instance: Grade reduction.
- Third instance: F grade and report to disciplinary committee.

-*-*-*- Do the assignment honestly; enjoy learning the course. -*-*-*-