Plan for Today

AST, CallGraph, and dynamic function stack
- example
- when do we know what?

Control Flow
- due to if stmts and while loops
- due to returns and exceptions

Exceptions in Java
- try and catch blocks
- finally blocks

Exceptions in MiniJava compiler

Example Program

```java
class Example {
    public static void main(String args[]) {
        foo();
        int retval = goo();
        if (retval == ERROR)
            System.out.println("goo ERROR in main");
    }

    void foo() {
        int retval = goo();
        if (retval == ERROR)
            System.out.println("goo ERROR in foo");
    }

    int goo() {
        if (random() > 42) return ERROR;
        return 0;
    }
}
```

Example Program with Exceptions

```java
class Example {
    public static void main(String args[]) {
        try {
            foo();
            goo();
        } catch (Exception e) {
            System.out.println("Caught in main");
        }

        void foo() {
            try {
                goo();
            } catch (Exception e) {
                System.out.println("Caught in foo");
            }
        }

        void goo() throws SomeException {
            if (random() > 42) throw new SomeException();
        }
    }
}
```

Exceptions in Java

```java
class Example {
    public static void main(String args[]) {
        PrintWriter out = null;
        try {
            out = new PrintWriter(new FileWriter("b.txt");
        } catch (IOException e) {
            System.err.println("Caught IOException: " + e.getMessage());
        } finally {
            if (out != null) {
                System.out.println("Closing PrintWriter");
                out.close();
            }
        }

        throw new MyException();
    }
}
```

Picture from Sun “The Java Tutorials”
Exception usage in the MiniJava compiler

try block
– is it necessary?
– applying the CheckTypes switch/visitor to the AST could result in a SemanticException being thrown

    try {
        // Create a lexer instance.
        Lexer lexer = new Lexer(new PushbackReader(
            new BufferedReader(new FileReader(filename)), 1024));
        ...
        // type checking
        ast.apply(new CheckTypes(globalST, linesToNodes));
    } catch (exceptions.SemanticException e) {
        System.err.println(e.getMessage());
        System.exit(2);
    } catch (Exception e) {
        e.printStackTrace();
        System.exit(1);
    }

But why isn’t SemanticException a checked exception?

Java rules (paraphrased)
– any exception that is not an Error or RuntimeException subclass must be caught
– if a method that doesn’t catch a checked exception it throws then it must indicate with a “throws” clause that it could throw that exception