1 Exercises

Exercise 1. What are the changes that you will need to make in the \textit{j--} code tree in order to support \textit{/} as the remainder operator on integers? For example, \(17 \div 5 = 3\).

Exercise 2. Write down the following class names in internal form:
- \texttt{java.util.ArrayList}
- \texttt{jminusminus.Parser}
- \texttt{Employee}

Exercise 3. Write down the JVM type descriptor for each of the following field/constructor/method declarations:
- private int \texttt{N};
- private String \texttt{s};
- public static final double \texttt{PI} = 3.141592653589793;
- public \texttt{Employee(String name){ ...}}
- public \texttt{Coordinates(double latitude, double longitude){ ...}}
- public \texttt{Object get(String key){ ...}}
- public \texttt{void put(String key, Object o){ ...}}
- public static \texttt{int[]} \texttt{sort(int[]} \texttt{n, boolean ascending){ ...}}
- public \texttt{int[][] transpose(int[] matrix){ ...}}

Exercise 4. Consider the following JVM bytecode for a \textit{j--} method \texttt{int mystery(int x, int y)}:

```
public static int mystery(int , int);
0: iconst_1
1: istore_2
2: iload_1
3: iconst_0
4: if_icmple 18
7: iload_2
8: iload_0
9: imul
10: iload_2
11: iload_1
12: iconst_1
13: isub
14: iload_1
15: goto 2
18: iload_2
19: ireturn
```

a. What does \texttt{mystery(2, 5)} return?
b. What does \texttt{mystery(3, 4)} return?
c. What does \texttt{mystery(x, y)} compute in general?

Exercise 5. Write a program \texttt{GenSquare.java} that produces, using \texttt{CLEmitter}, a \texttt{Square.class} program, which accepts an integer \texttt{n} as command-line argument and prints the square of that number as output.

Exercise 6. A variant of the previous exercise is one in which you are given a program such as \texttt{GenSquare.java} and asked what it computes.
2 Solutions

Solution 1.

Lexical Grammar

DIV ::= "/"  

Multiplicative Expression

multiplicativeExpression ::= unaryExpression  
{ ( STAR | DIV ) unaryExpression }

Semantics

JBinaryExpression:  
  - JDivideOp  
  - lhs and rhs must be integers.

enum TokenKind {  
  DIV("/"),
}

Scanner.java

```java
if (ch == '/') {  
  nextCh();  
  if (ch == '/') {  
    // CharReader maps all new lines to '\n'.  
    while (ch != '\n' && ch != EOFCH) {  
      nextCh();  
    }  
  } else {  
    return new TokenInfo(DIV, line);  
  }
}
```

Parser.java

```java
private JExpression multiplicativeExpression() {  
  int line = scanner.token().line();  
  boolean more = true;  
  JExpression lhs = unaryExpression();  
  while (more) {  
    if (have(STAR)) {  
      lhs = new JMultiplyOp(line, lhs, unaryExpression());  
    }  
    else if (have(DIV)) {  
      lhs = new JDivideOp(line, lhs, unaryExpression());  
    }  
    else {  
      more = false;  
    }
  }  
  return lhs;
}
```

JBinaryExpression.java

```java
class JDivideOp extends JBinaryExpression {  
  public JDivideOp(int line, JExpression lhs, JExpression rhs) {  
    super(line, "/", lhs, rhs);  
  }
  
  public JExpression analyze(Context context) {  
    lhs = (JExpression) lhs.analyze(context);  
    rhs = (JExpression) rhs.analyze(context);  
    lhs.type().mustMatchExpected(line, Type.INT);  
    rhs.type().mustMatchExpected(line, Type.INT);  
    type = Type.INT;  
    return this;
  }
  
  public void codegen(CLEmitter output) {  
    lhs.codegen(output);
  }
}
```
Compilation

```java
rhscodegen(output);
output.addNoArgInstruction(IDIV);
}
}
```

Solution 2.

- java/util/ArrayList
- jminusminus/Parser
- Employee

Solution 3.

- I
- Ljava/lang/String;
- D
- (Ljava/lang/String;)V
- (D)V
- (Ljava/lang/String;)Ljava/lang/Object;
- (Ljava/lang/String;Ljava/lang/Object;)V
- ((II)I
- ((II)(I

Solution 4.

- 32
- 81
- $x^y$

Solution 5.

```java
import jminusminus.CLEmitter;
import static jminusminus.CLConstants.*;
import java.util.ArrayList;

class GenSquare {
    public static void main(String[] args) {
        CLEmitter e = new CLEmitter(true);
        ArrayList<String> accessFlags = new ArrayList<String>();
        accessFlags.add("public");
        e.addClass(accessFlags, "Square", "java/lang/Object", null, true);
        accessFlags.clear();
        accessFlags.add("public");
        accessFlags.add("static");
        e.addMethod(accessFlags, "main", "([Ljava/lang/String;)V", null, true);
        e.addNoArgInstruction(ALOAD_0);
        e.addNoArgInstruction(ALOAD_0);
    }
}
```
Compilation

Solution 6. GenSquare.java uses the CLEmitter library to generate a program called Square.class, which accepts an integer n as command-line argument and prints the square of that number as output.