INF2100

Løsningsforslag til oppgaver 3.–15. oktober 2007

Oppgave 1

Her er det mange mulige løsninger — i figur 1 på neste side er én:

Oppgave 2

```
   void print (int level) {
      for (int i = 1;  i <= level;  ++i)
         System.out.print("    ");
      System.out.println(name);
      if (first_child != null)
         first_child.print(level+1);
      if (next != null)
         next.print(level);
   }
```

Oppgave 3

Her er det også mange mulige løsninger — her er én:

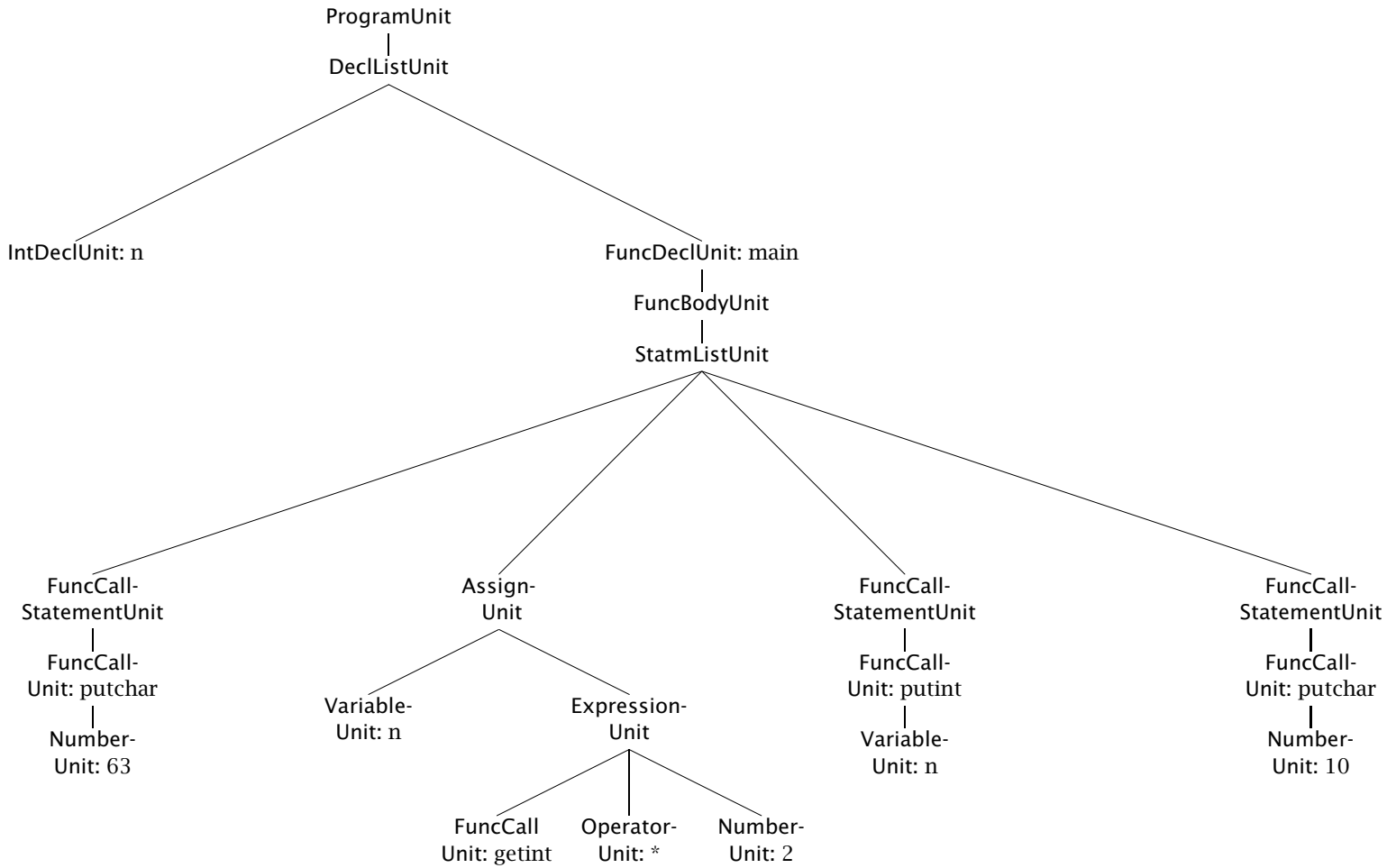
```
import java.io.*;

class E {
   public static void main(String arg[]) {
      Scanner.init();  Scanner.readNext();

      Program p = Program.parse();
      if (Scanner.curToken != Token.eofToken)
         Error.error("Syntax error: Illegal "+Scanner.curToken);

      System.out.println("The value is "+p.eval());
   }
}

class Program {
   Expression e;
```



Figur 1: Ett mulig svar på oppgave 1

```
    public static Program parse() {
        Program p = new Program();
        p.e = Expression.parse();
        return p;
    }

    public int eval() {
        return e.eval();
    }
}

class Expression {
    ExprElem e;

    public static Expression parse() {
        Expression ex = new Expression();
        ExprElem f = Factor.parse();
        while (Scanner.curToken==Token.plusToken ||
               Scanner.curToken==Token.minusToken) {
            Operator o = Operator.parse();
            o.param1 = f;  o.param2 = Factor.parse();
            f = o;
        }
        ex.e = f;
        return ex;
    }

    public int eval() {
        return e.eval();
    }
}

abstract class ExprElem {
    public abstract int eval();
}

class Factor extends ExprElem {
    Number n;

    public static Factor parse() {
        Factor f = new Factor();
        f.n = Number.parse();
        return f;
    }

    public int eval() {
        return n.eval();
    }
}
```

```
class Number {
   int val;

   public static Number parse() {
      Number n = new Number();
      if (Scanner.curToken != Token.numberToken)
         Error.error("Found not a number but "+Scanner.curToken);
      n.val = Scanner.curNumber;
      Scanner.readNext();
      return n;
   }

   public int eval() {
      return val;
   }
}

class Operator extends ExprElem {
   Token op;
   ExprElem param1, param2;

   public static Operator parse() {
      Operator o = new Operator();
      o.op = Scanner.curToken;
      Scanner.readNext();
      return o;
   }

   public int eval() {
      if (op == Token.plusToken)
         return param1.eval()+param2.eval();
      else
         return param1.eval()-param2.eval();
   }
}

enum Token { numberToken, plusToken, minusToken, eofToken, noToken }

class Scanner {
   public static Token curToken;
   public static int curNumber;

   private static LineNumberReader f;

   public static void init() {
      f = new LineNumberReader(new InputStreamReader(System.in));
   }
```

```
    public static void readNext() {
        curToken = Token.noToken;
        while (curToken == Token.noToken) {
            int c = '?';
            try {
                c = f.read(); // Read one character
            } catch (IOException e) {
                Error.error("Read error!");
            }

            if (c < 0) {
                curToken = Token.eofToken;
            } else if (c == '+') {
                curToken = Token.plusToken;
            } else if (c == '-') {
                curToken = Token.minusToken;
            } else if (Character.isDigit(c)) {
                curToken = Token.numberToken;  curNumber = c-'0';
            } else if (Character.isWhitespace(c)) {
                // Ignore space
            } else {
                Error.error("Illegal character: '"+(char)c+"'!");
            }
        }
        // For testing:
        // System.out.println("Scanner: Read a "+curToken);
    }
}

class Error {
    static void error(String message) {
        System.out.println("ERROR: "+message);
        System.exit(1);
    }
}
```