Aflevering 1

Studerende 1, 2017xxxxx   Studerende 2, 2017xxxxx

X. YYYY 20ZZ

Opgave 9

Kode

```scala
def eval(e: Exp): Int = e match {
  case IntLit(c) => c
  case BinOpExp(leftexp, op, rightexp) =>
    val leftval = eval(leftexp)
    val rightval = eval(rightexp)
    op match {
      case PlusBinOp() => leftval + rightval
      case MinusBinOp() => ???
      case MultBinOp() => ???
      case DivBinOp() =>
        if (rightval == 0)
          throw new InterpreterError(s"Division by zero", e)
        leftval / rightval
      case ModuloBinOp() => ???
      case MaxBinOp() =>
        if (???) ??? else ???
    }
  case UnOpExp(op, subexp) =>
    val subexpval = eval(subexp)
    op match {
      case NegUnOp() => -subexpval
    }
}
```

Beskrivelse

Trace-mekanismen fungerer ved ...

Når fortolkeren køres med argumenterne `-run -trace examples/calc1.s` fås ...

Opgave 10

Kode

```scala
def unparse(e: AstNode): String =
  ???
```

Beskrivelse

Unparse-mekanismen fungerer ved ...

Opgave 11

Kode

```scala
/**
  * Expressions.
  */
sealed abstract class Exp extends AstNode

case class BinOpExp(leftexp: Exp, op: BinOp, rightexp: Exp) extends Exp

case class UnOpExp(op: UnOp, exp: Exp) extends Exp

case class IntLit(c: Int) extends Exp

/**
  * Binary operators.
  */
sealed abstract class BinOp

case class PlusBinOp()extends BinOp

case class MinusBinOp()extends BinOp

case class MultBinOp()extends BinOp

case class DivBinOp()extends BinOp

case class ModuloBinOp()extends BinOp

case class MaxBinOp()extends BinOp

/**
  * Unary operators.
  */
sealed abstract class UnOp

case class NegUnOp() extends UnOp
```

Beskrivelse

I den objekt-orienterede stil ...