Opgaveløsninger (sæt 1)

Opgave 1 (2.6)

Hurtig version (se lærebogen nederst side 9):

```
u = 12345, v = 56789
u = 56789, v = 12345
u = 7409, v = 12345
u = 12345, v = 7409
u = 4936, v = 7409
u = 7409, v = 4936
u = 2473, v = 4936
u = 4936, v = 2473
u = 2463, v = 2473
u = 2473, v = 2463
u = 10, v = 2463
u = 2463, v = 10
u = 3, v = 10
u = 10, v = 3
u = 1, v = 3
u = 3, v = 1
u = 0, v = 1
```

Opgave 2

```
class Fraction {
    Fraction(int a, int b) {
        if (b == 0)
            throw new RuntimeException(
                    "Fraction: 0 in denominator");
        if (a == 0) { this.b = 1; return; }
        int d = gcd(Math.abs(a), Math.abs(b));
        this.a = a/d; this.b = b/d;
   }
   Fraction(int a) { this.a = a; this.b = 1; }
   public int numerator() { return a; }
   public int denominator() { return b; }
   public Fraction add(Fraction f)
     { return new Fraction(a*f.b+b*f.a, b*f.b); }
   public Fraction subtract(Fraction f)
     { return new Fraction(a*f.b-b*f.a, b*f.b); }
   public Fraction multiply(Fraction f)
     { return new Fraction(a*f.a, b*f.b); }
   public Fraction divide(Fraction f)
     { return new Fraction(a*f.b, b*f.a); }
   public String toString() {
        return a == 0 || b == 1 ? "" + a :
               ((a > 0) != (b > 0) ? "-" : "")
               + Math.abs(a) + "/" + Math.abs(b);
   }
   private int a, b;
   private int gcd(int u, int v) {
       while (u > 0) {
           if (u < v) { int t = u; u = v; v = t; }
           u %= v;
       }
       return v;
   }
}
```

Nedenstående hovedprogram udskriver summen af 1/i for i = 1 til i = 10.

```
        public class Program {
            public static void main(String args[]) {
                Fraction sum = new Fraction(0);
                for (int i = 1; i <= 10; i++)
                    sum = sum.add(new Fraction(1, i));
                System.out.println(sum);
           }
        }
```

En kørsel gav følgende (korrekte) udskrift:

```
7381/2520
```

Opgave 3

```
import java.math.BigInteger;

class Fraction {
    Fraction(BigInteger a, BigInteger b) {
        if (a == null)
            throw new RuntimeException(
                    "Fraction: numerator is null");
        if (b == null)
            throw new RuntimeException(
                    "Fraction: denominator is null");
        if (b.compareTo(ZERO) == 0)
            throw new RuntimeException(
                    "Fraction: 0 in denominator");
        if (a.compareTo(ZERO) == 0)
            { this.a = ZERO; this.b = ONE; return; }
        BigInteger d = a.gcd(b);
        this.a = a.divide(d); this.b = b.divide(d);
    }
    Fraction(BigInteger a) { this(a, ONE); }
    Fraction(long a, long b)
      { this(BigInteger.valueOf(a), BigInteger.valueOf(b)); }
    Fraction(long a) { this(a, 1); }
    public BigInteger numerator() { return a; }
    public BigInteger denominator() { return b; }
    public Fraction add(Fraction f) {
        return new Fraction(a.multiply(f.b).add(b.multiply(f.a)),
                            b.multiply(f.b));
    }
    public Fraction subtract(Fraction f) {
       return new Fraction(a.multiply(f.b).subtract(
                           b.multiply(f.a)), b.multiply(f.b));
    }
    public Fraction multiply(Fraction f)
      { return new Fraction(a.multiply(f.a), b.multiply(f.b)); }
    public Fraction divide(Fraction f)
      { return new Fraction(a.multiply(f.b), b.multiply(f.a)); }
    public String toString() {
        return a.compareTo(ZERO) == 0 || b.compareTo(ONE) == 0 ?
               "" + a :
               ((a.compareTo(ZERO) > 0) != (b.compareTo(ZERO) > 0)
                ? "-" : "") + a.abs() + "/" + b.abs();
    }
    private final static BigInteger ZERO = BigInteger.valueOf(0),
                                    ONE =  BigInteger.valueOf(1);
    private BigInteger a, b;
}
```

Nedenstående hovedprogram udskriver summen af 1/i for i = 1 til i = 100.

```
public static void main(String args[]) {
    Fraction sum = new Fraction(0);
    for (int i = 1; i <= 100; i++)
        sum = sum.add(new Fraction(1, i));
    System.out.println(sum);
}
```

En kørsel gav følgende (korrekte) udskrift:

14466636279520351160221518043104131447711/2788815009188499086581352357412492142272