Eidgenössische Technische Hochschule Zürich
Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Swiss Federal Institute of Technology Zurich

Institut für theoretische Informatik
Dr. B. Gärtner

10. Januar 2005

Informatik für Mathematiker und Physiker Serie 11 WS 04/05

URL: `http://www.ti.inf.ethz.ch/ew/courses/Info1_04/`

Aufgabe 1 [Schnellübung – 20 Min.] (5 Punkte)

Was passiert in der untenstehenden Funktion `foo`? Beschreiben Sie ihre Funktionalität, indem Sie Pre– und Postconditions angeben. Erläutern Sie Ihre Angaben.

Gruppe A–H

```
typedef std::vector<int>::const_iterator Cit;

Cit foo(Cit b, Cit e, Cit& x)
{
  if (b == e) {
    x = e;
    return e;
  }
  Cit c = b;
  Cit y = foo(++c, e, x);
  if (y == e) return b;
  if (*b > *y) {
    x = y;
    return b;
  }
  if (x == e || *b > *x) x = b;
  return y;
}
```

Gruppe I

```
typedef std::vector<int>::const_iterator Cit;

Cit foo(Cit b, Cit e, Cit& max)
{
  if (b == e) return e;
  It n = b++;
  if (b == e) {
    max = n;
    return n;
  }
  It m = b;
  if (*n > *m) {
    m = n;
    n = b;
  }
  It c = foo(++b, e, max);
  if (c == e) {
    max = m;
    return n;
  }
  if (*m > *max) max = m;
  if (*n < *c) return n;
  return c;
}
```

Gruppe J und K

```
typedef std::vector<int>::const_iterator Cit;

int foo(Cit b, Cit e, int& g)
{
  if (b == e) {
    g = 0;
    return 0;
  }
  int z = *b;
  int u = foo(++b, e, g);
  if (z % 2 == 0)
    g += z;
  else
    u += z;
  return u;
}
```

Aufgabe 2 (4 Punkte)

Betrachten Sie folgendes Programm-Fragment. Geben Sie für jeden Funktionsaufruf an, welche Funktion jeweils aufgerufen wird und begründen Sie Ihre Antwort.

```
void foo(double, double)      { ... }  // Funktion A
void foo(unsigned int, int)   { ... }  // Funktion B
void foo(float, unsigned int) { ... }  // Funktion C
```

a) `foo(1, 1)`

b) `foo(1u, 1.0f)`

c) `foo(1.0, 1)`

d) `foo(1, 1u)`

e) `foo(1, 1.0f)`

f) `foo(1.0f, 1.0)`

Aufgabe 3 (3 Punkte)

Beweisen Sie, dass der Quicksort Algorithmus aus Vorlesung 10 bei Eingabe von n Elementen höchstens $\binom{n}{2}$ Vergleiche durchführt.

Abgabe: Aufgaben 2+3: am 18. Januar 2005, in der Pause der Vorlesung, schriftlich.

Eidgenössische Technische Hochschule Zürich
Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Swiss Federal Institute of Technology Zurich

Institut für theoretische Informatik
Dr. B. Gärtner

10. Januar 2005

Informatik I: Material aus der Vorlesung

Programm: rational.C

```
// Programm: rational.C
// Brueche ganzer Zahlen.

// Achtung: Programm compiliert so nicht!

namespace ifm {

  // Klasse zur Repraesentation von Bruechen ganzer Zahlen.
  class Rational {
  public:

    typedef int NT;

    Rational(const NT& n, const NT& d);
    // PRE: d != 0.
    // POST: Bruch initialisiert als n / d.

    Rational(const NT& n);
    // POST: Bruch initialisiert als n / 1.

    const NT& n() const;
    // POST: Rueckgabewert ist Zaehler des Bruchs.

    const NT& d() const;
    // POST: Rueckgabewert ist Nenner des Bruchs.

    void normalize();
    // POST: Zaehler und Nenner sind teilerfremd.

    Rational& operator+=(const Rational& x);
    // POST: x wurde zu *this addiert.

    Rational& operator-=(const Rational& x);
    // POST: x wurde von *this subtrahiert.

  private:

    NT n_; // numerator (Zaehler)
    NT d_; // denominator (Nenner) Invariante: d_ > 0.
  };

  Rational operator+(const Rational& x, const Rational& y)
  // POST: Rueckgabewert ist x + y.
  {
    Rational z = x;
    z += y;
    return z;
  }

  Rational operator-(const Rational& x, const Rational& y)
  // POST: Rueckgabewert ist x - y.
  {
    Rational z = x;
    z -= y;
    return z;
  }

  // ...

} // namespace ifm

int main()
{
  ifm::Rational q(1, 2);
  ifm::Rational r(2, 3);
  ifm::Rational x = q + r;
  ifm::Rational y = q - r;

  std::cout << "q   = " << q.n() << "/" << q.d() << "\n"
            << "r   = " << r.n() << "/" << r.d() << "\n"
            << "q+r = " << x.n() << "/" << x.d() << "\n"
            << "q-r = " << y.n() << "/" << y.d() << std::endl;
  return 0;
}
```