BERGISCHE UNIVERSITÄT
GESAMTHOCHSCHULE WUPPERTAL
GAUSS-STRASSE 20
42097 WUPPERTAL
(Korrespondenzanschrift)
42119 WUPPERTAL
(Lieferanschrift)
TELEX 8 592 262 bughw
TELEFAX (0202) 439-2901
TELEFON (0202) 439-1

Fachbereich 7
MATHEMATIK

Prof. Dr. Hans-Jürgen Buhl
Praktische Informatik / Numerik

e-mail: Juergen.Buhl@math.uni-wuppertal.de

Einführung in die Informatik und Programmierung (Informatik I)

WS2000/2001 – Übungsblatt 14 (optional)

7. Februar 2001
Bearbeitungstermin: 7. KW

Aufgabe 1. *Sieb des Eratosthenes, 4 Punkte*
Eine weitere Art, die ersten Primzahlen bis zu einer gegebenen natürlichen Zahl `N` zu berechnen, geht folgendermaßen vor:

- Man schreibe alle Zahlen von `2` bis `N` in eine Tabelle.
- Man markiere die Zahl `2` und streiche dann alle Vielfachen von `2` aus der Tabelle.
- Ist `n` die kleinste nicht gestrichene und nicht markierte Zahl, so markiere man `n` und streiche alle Vielfachen von `n`.
- Man wiederhole den vorigen Schritt für alle solchen `n` mit $\mathtt{n} \leq \sqrt{\mathtt{N}}$.

Die Primzahlen bis `N` sind dann die in der Tabelle markierten beziehungsweise nicht gestrichenen Zahlen.

Programmieren Sie den Algorithmus inklusive Zeitvergleich.

Verbessern Sie ihn, indem Sie eine Tabelle nur der ungeraden Zahlen benutzen (Zeitvergleich zu den schon bekannten drei Versionen der Primzahlberechnung).

Aufgabe 2. *Vektoren mit dynamischer Dimension, 4 Punkte*
Um erst zur Laufzeit die aktuelle Dimension von Vektoren festzulegen, muß man folgendermaßen vorgehen:

```
////////////////////////////////
// Datei:   dynvec.cc
// Version: 1.0
// Zweck:   dynamisch dimensionierte Vektoren
// Autor:   Hans-Juergen Buhl
// Datum:   28.09.1998
////////////////////////////////

#include <iostream>

using namespace std;

int main()
{
 try{
  int n;
  double* v;  // Variable für Startadresse für dyn. erzeugte Vektoren

  cout << "Bitte geben Sie die Dimension (ganze Zahl größer 0) ein: ";
  cin >> n;
  if (n <= 0) throw "n ist keine ganze Zahl größer 0";

  v = new double[n];
  for (int i=0; i < n; i++)
    v[i] = 2.0 * i;

  for (int j=0; j < n; j++)
    cout << "v[" << j << "] == " << v[j] << endl;

  delete []v;

  return 0;
 }
 catch(const char* err){
    cerr << endl << err << endl;
    return 1;
 }
}
```

Testen Sie das Programm. Ändern Sie das Programm wie folgt ab:

```
  ...
  int n;
  double* v;  // Variable für Startadresse für dyn. erzeugte Vektoren
```

```
cout << v[0] << endl;

cout << "Bitte geben Sie die Dimension (ganze Zahl größer 0) ein: ";
 ...
```

Welche Fehlermeldung erscheint bei der Übersetzung und was hat sie zu bedeuten? Was geschieht beim Programmlauf?

Ändern Sie das Programm erneut ab:

```
...
int n;
double* v(0);  // Variable für Startadresse für dyn. erzeugte Vektoren

cout << v[0] << endl;

cout << "Bitte geben Sie die Dimension (ganze Zahl größer 0) ein: ";
...
```

Wie ändert sich das Programmverhalten? Warum sollte eine solche Initialisierung durch den Wert 0 immer durchgeführt werden? Leider können Dereferenzierungen des Nullpointers nicht wie Exceptions abgefangen werden. In der Informatik II werden Sie eine Technik namens "smart pointers" kennenlernen, die für diese Problematik eine geeignete Lösung darstellt.

Ändern Sie das Programm wiederum ab:

```
#include <exception>
...
int n;
double* v(0);  // Variable für Startadresse für dyn. erzeugte Vektoren

cout << "Bitte geben Sie die Dimension (ganze Zahl größer 0) ein: ";
...
delete []v;

return 0;
}
catch(const char* err){
   cerr << endl << err << endl;
   return 1;
}
catch(const exception& e){
   cerr << endl << e.what() << endl;
   return 2;
}
```

Versuchen Sie jetzt ein Feld der Dimension 100000000 zu erzeugen. Was stellen Sie fest?

Aufgabe 3. *Vektoren mit dynamischer Dimension (Forts.), 4 Punkte*
Die Lebenszeit von mittels `new` erzeugten dynamischen Objekten ist nicht an die bisher bekannte Block-Regel gebunden; dynamische Objekte existieren solange (und belegen solange kostbaren Hauptspeicherplatz), bis sie explizit mittels `delete` wieder freigegeben werden.
Was passiert in einer Version der folgenden Art:

```
...
v = new double[n];
for (i=0; i < n; i++)
  v[i] = 2.0 * i;

for (j=0; j < n; j++)
  cout << "v[" << j << "] == " << v[j] << endl;

v = new double[15];

for (i=0; i < 15; i++)
  v[i] = 2.0 * i;

for (j=0; j < 15; j++)
  cout << "v[" << j << "] == " << v[j] << endl;


delete []v;
...
```

Starten Sie dazu den Debugger (`workshop -D dynvec`), wählen Sie unter „Checks“ den Punkt „Enable Memuse Checking“ an, und starten Sie das Programm mit dem „run“-Knopf:
Was hat die erscheinende Fehlermeldung

```
Actual leaks report    (actual leaks:         1  total size:     272 bytes)
 Total  Num of  Leaked      Allocation call stack
 Size   Blocks  Block
                Address
======  ====== ==========  ======================================
   272       1    0x212d0   operator new<-main


Possible leaks report  (possible leaks:       0  total size:       0 bytes)
```

zu bedeuten?
Wie können Sie dieses Speicherleck vermeiden? Testen Sie!

Aufgabe 4. *GUI-Programmierung: Zeichnen, 4 Punkte*
Betrachten Sie das folgende Programm:

```
#include <qapplication.h>
#include <qwidget.h>
#include <qpainter.h>
#include <math.h>

//  g++ qt-graph2.cc -o qt-graph2 -lqt
//  mit Beschriftung

class Bild : public QWidget
{
public:
        void paintEvent( QPaintEvent * )
        {
                int hh = height()/2;
                int b  = width();
                double x, h(10.0/b);

                QPainter p( this );

                for(int i = 0; i < b; i++)
                {
                        x = i * h;
                        p.lineTo(i, (-sin(x)*hh)+hh);
                }

                QString s("sin(x)");
                p.drawText(30, 220, s);

        };
};

int main( int argc, char **argv )
{
        QApplication a( argc, argv );

        Bild hello;

        hello.resize( 300, 300 );
        hello.setPalette(QPalette(QColor(255, 255, 255)));

        a.setMainWidget( &hello );
        hello.show();

        return a.exec();
}
```

Übersetzen Sie es mittels `g++ qt-graph2.cc -o qt-graph2 -lqt` und bringen Sie es zur Ausführung.
Schreiben Sie ein ähnliches Programm, das die Funktion `sin(x) * exp(-x)` im Intervall `[0.0, 10.0]` zeichnet. Vergessen Sie eine Beschriftung der Zeichnung und das Zeichnen des Koordinatensystems nicht.

Aufgabe 5. *formatierte Textausgabe, 4 Punkte*
Übersetzen Sie das Programm:

```
#include <qapplication.h>
#include <qwidget.h>
#include <qbrush.h>
#include <qvbox.h>
#include <qhbox.h>
#include <qpushbutton.h>
#include <qtextview.h>

#include <string>

//
//  g++ qt-graph3.cc -o qt-graph3 -lqt
//
//                Text-Ausgabe

class TextFenster : public QVBox
{
protected:

        QTextView* view;
        QPushButton* bClose;

public:
        TextFenster(const QString inhalt,
                    QWidget* parent = 0, const char* name = 0)
        : QVBox(parent, name){

        setMargin(5);


        view = new QTextView(this);
        QBrush paper;
        paper.setPixmap(QPixmap("marble.png"));
        view->setPaper(paper);
        view->setText(inhalt);
        view->setMinimumSize(450, 250);

        QHBox *buttons = new QHBox(this);
        buttons->setMargin(5);

        bClose = new QPushButton("&Close", buttons),
```

```
        connect(bClose, SIGNAL(clicked()), qApp, SLOT(quit()));

        }

};

int main( int argc, char **argv )
{
        QApplication a(argc, argv);

        static QString Text(
                "<center>"
                "<b><br>Der Trichter</b><br>"
                "<br>"
                "Zwei Trichter wandeln durch die Nacht.<br>"
                "Durch ihres Rumpfs verengten Schacht<br>"
                "fließt weißes Mondlicht<br>"
                "still und heiter<br>"
                "auf ihren<br>"
                "Waldweg<br>"
                "u. s.<br>"
                "w.<br>"
                "</center>"
              );

        TextFenster win(Text, 0, 0);

        win.resize(450, 250);
        win.setCaption("Christian Morgenstern:");

        a.setMainWidget( &win );
        win.show();

        return a.exec();
}
```

Schreiben Sie eine ähnliche Version, die zur Ausgabe beliebiger Fehlermeldungen (aus einem statischen Feld von Zeichenketten) geeignet ist und testen Sie.
Bemerkung: Neben der Datei `qt-graph3.cc` benötigen Sie auch die Datei `marble.png`, die einen „schönen“ Fensterhintergrund enthält!