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

Betriebssysteme: Konzepte, Dienste, Schnittstellen (Betriebssysteme und betriebssystemnahe Programmierung)

SS 2003 – Übungsblatt 5

18. Juni 2003
Ausgabe: 4. Juni 2003

Aufgabe 1. *stty und system()*

Übersetzen Sie das folgende kleine Programm

```
////////////////////////////////
// Datei:   inchar.cc
// Version: 1.0
// Autor:   Hans-Juergen Buhl
// Datum:   3. 6. 2003
////////////////////////////////

#include        <iostream>
#include        <string>
#include        <cassert>

#include        <cstdlib>

using namespace std;

int main()
{
  char inp('\000');

 system("clear");
 cout << endl << endl << "Please press any key: ";

 system("stty raw -echo");
 cin >> inp;
 system("stty sane echo");
```

```
 cout << endl << endl
      << " Die Taste mit dem Code " << static_cast<int>(inp)
      << " wurde betätigt" << endl << endl;
 cout << "Ende\n" << endl << endl;

}
```

und informieren Sie sich mittels `man` über die benutzten Optionen des Kommandos `stty`. Beschreiben Sie dann die Wirkungsweise des Programms. Was geschieht, wenn Sie den Aufruf von `system("stty sane echo")` vergessen?

Aufgabe 2. *isatty*

Bringen Sie das folgende Programm zum Ablauf

```
#include <termios.h>
#include <unistd.h>

int
main(void)
{
struct termios term;
long vdisable;

if (isatty(STDIN_FILENO) == 0){
perror("standard input is not a terminal device");
exit(1);
}

if ( (vdisable = fpathconf(STDIN_FILENO, _PC_VDISABLE)) < 0){
perror("fpathconf error or _POSIX_VDISABLE not in effect");
exit(1);
}

if (tcgetattr(STDIN_FILENO, &term) < 0){
/* fetch tty state */
perror("tcgetattr error");
exit(1);
}

term.c_cc[VINTR] = vdisable; /* disable INTR character */
term.c_cc[VEOF]  = 2; /* EOF is Control-B */

if (tcsetattr(STDIN_FILENO, TCSAFLUSH, &term) < 0){
perror("tcsetattr error");
exit(1);
}
```

```
exit(0);
}
```

und erklären Sie es Zeile für Zeile. Überprüfen Sie dessen Wirkungsweise mit Hilfe des Kommandos `stty -a`. Wie kann man den gleichen Effekt interaktiv mit Hilfe von `stty` erzeugen?

Aufgabe 3. *winsize*

Bringen Sie das folgende Programm zum Ablauf

```
#include <signal.h>
#include <termios.h>
#ifndef TIOCGWINSZ
#include <sys/ioctl.h>/* 44BSD requires this too */
#endif
#include <unistd.h>

static void pr_winsize(int), sig_winch(int);

int
main(void)
{
if (isatty(STDIN_FILENO) == 0)
exit(1);

if (signal(SIGWINCH, sig_winch) == SIG_ERR){
perror("signal error");
}

pr_winsize(STDIN_FILENO);   /* print initial size */
for ( ; ; )                 /* and sleep forever */
pause();
}

static void
pr_winsize(int fd)
{
struct winsize size;

if (ioctl(fd, TIOCGWINSZ, (char *) &size) < 0){
                perror("signal error");
        }

printf("%d rows, %d columns\n", size.ws_row, size.ws_col);
}

static void
sig_winch(int signo)
{
```

```
printf("SIGWINCH received\n");
pr_winsize(STDIN_FILENO);
return;
}
```

und erklären Sie seine Wirkungsweise Zeile für Zeile.

Aufgabe 4. *SystemAndDate*

Arbeiten Sie analog mit:

```
////////////////////////////////
// Datei:   SystemAndDate.cc
// Autor:   Hans-Juergen Buhl
// Datum:   3. 6. 2003
//
// Bemerkung: evtl. explizit mit -lnsl uebersetzen
//
////////////////////////////////

#include <iostream>
#include <string>

#include <time.h>               // oder ctime
#include <locale.h>             // oder clocale

#include <stdio.h>              // oder cstdio

#include <unistd.h>
#include <netdb.h>

using namespace std;

class Nachricht {

  const string Text;

  string CreateTime;
  string Hostname;
  string Domainname;
  string Username;

public:

  Nachricht(const string& t);

  void print() const { cout << Text; };

  void printSignature() const { cout << "Signatur: " << CreateTime
                                << "          " << Username << "@"
```

```
                                    << Hostname << "." << Domainname; };

};

Nachricht::Nachricht(const string& t) : Text(t), Username(cuserid(NULL))
{
    time_t Now = time(NULL);
    setlocale(LC_ALL, "de");
    CreateTime = ctime(&Now);

    char host[20];
    gethostname(host, sizeof(host));
    Hostname = host;

    // char domain[40];
    // getdomainname(domain, sizeof(domain)); //funktioniert nicht!warum?
    {
struct hostent *ho;
ho = gethostbyname(host);
// clog << ho->h_name << endl;
char *domain = strchr(ho->h_name, '.');
if (domain != 0) domain++;
// clog << (long)domain << endl;

    if (domain==0)
Domainname="";
    else
Domainname = domain;
    }
}

int main()
{
  Nachricht N1("... Meine erste automatisch signierte Nachricht ...");

  N1.print();
  cout << endl << endl;

  N1.printSignature();
  cout << endl;

  return 0;
}
```

Ergänzen Sie abschließend überall dort, wo nötig, die Abfrage des Erfolgszustandes der aufgerufenen Funktionen.

Aufgabe 5. *ddd auf Fehlersuche*

Das folgende Programm enthält einige gravierende Fehler. Mittels des Debuggers ddd können diese bequem genauer untersucht werden:

```
////////////////////////////////
// Datei:   vektor1.cc
// Version: 1.0
// Zweck:   Vektor als Klasse
// Autor:   Hans-Juergen Buhl
// Datum:   26.01.99
////////////////////////////////

#include <iostream>

using namespace std;

class vektor{

  const int low;        // v(low..high)
  const int high;

  double* v;            // Startadresse fuer dyn. verwaltetes Exemplar

public:

  vektor(int h, int l = 1, double d = 0.0);
                                                //  v(l..h) = d

  vektor(const double x[], int n);              // v(1..n) = x[0..n-1]

  ~vektor(){ delete []v; };

  double& operator()(int i);
  double  operator()(int i) const;

  int lo() const { return low;  };
  int hi() const { return high; };

  friend ostream& operator<<(ostream& os, const vektor& v);

};

vektor::vektor(int h, int l, double d) : low(l), high(h)
{
  int size(h-l+1);
  if (size < 1) throw "falsche Vektor-Länge in Konstruktor";
  v = new double[size];
  if (v == 0) throw "kein freier Speicherplatz mehr verfügbar";
  for (int j=0; j < size; j++)
```

```
    v[j] = d;
};

vektor::vektor(const double x[], int n) : low(1), high(n)
{
  if (n < 1) throw "falsche Vektor-Länge in Konstruktor";
  // ...    z u    e r g ä n z e n
};

double& vektor::operator()(int i)
{
  if ( (i < low) || (i > high))
     throw "Indexverletzung bei Komponentenzugriff";
  return v[i-low];
};

double  vektor::operator()(int i) const
{
  if ( (i < low) || (i > high))
     throw "Indexverletzung bei Komponentenzugriff";
  return v[i-low];
};

ostream& operator<<(ostream& os, const vektor& w)
{
  os << "( ";
  os << w(w.lo());
  for (int i=w.lo()+1; i <= w.hi(); i++)
    os << " , " << w(i);
  os << " )";
  return os;
};

int main()
{
  vektor y(5, 1, 3.0);
  cout << y.lo() << " " << y.hi() << endl;
  cout << y  << endl;
  for (int i=y.lo(); i <= y.hi(); i++)
    y(i) = i*2;
  cout << y  << endl;

  vektor x(8, 2);
  cout << x.lo() << " " << x.hi() << endl;
  cout << x  << endl;
  for (int k=x.lo(); k <= x.hi(); k++)
    x(k) = k*2;
  cout << x << endl;
```

```
  double zh[] = { 1.0, 3.0, 2.0, 4.0 };
  vektor z(zh, 4);
  cout << z.lo() << " " << z.hi() << endl;
  for (int j=z.lo(); j <= z.hi(); j++)
    z(j) *= z(j);
  cout << z << endl;

  return 0;
}
```

Nach abnormalem Programmabbruch

```
./vektor1
1 5
( 3 , 3 , 3 , 3 , 3 )
( 2 , 4 , 6 , 8 , 10 )
2 8
( 0 , 0 , 0 , 0 , 0 , 0 , 0 )
( 4 , 6 , 8 , 10 , 12 , 14 , 16 )
1 4
( 16 , 0 , 16.5493 , 0 )
Segmentation fault
```

und Überprüfung mittels

```
g++ -g vektor1.cc   -o vektor1
gdb vektor1
(gdb) run
Starting program: /home/buhl/vektor1
1 5
( 3 , 3 , 3 , 3 , 3 )
( 2 , 4 , 6 , 8 , 10 )
2 8
( 0 , 0 , 0 , 0 , 0 , 0 , 0 )
( 4 , 6 , 8 , 10 , 12 , 14 , 16 )
1 4
( 16 , 0 , 16.5493 , 0 )

Program received signal SIGSEGV, Segmentation fault.
0x40184d2e in _int_free () from /lib/libc.so.6
gdb) bt
#0  0x40184d2e in _int_free () from /lib/libc.so.6
#1  0x40183a5f in free () from /lib/libc.so.6
#2  0x400b76b1 in operator delete(void*) () from /usr/lib/libstdc++.so.5
#3  0x400b770d in operator delete[](void*) () from /usr/lib/libstdc++.so.5
#4  0x0804908b in ~vektor (this=0xbffff6c0) at vektor1.cc:27
#5  0x08048f54 in main () at vektor1.cc:100
#6  0x401228ae in __libc_start_main () from /lib/libc.so.6
(gdb) q
```

ist man nur wenig schlauer. Ändern Sie das Programm, um die Fehlerstelle genauer zu untersuchen. Benutzen Sie `ddd` statt `gdb` zur schrittweisen (Knopf `Next` bzw. `Step`) Programmausführung. Beheben Sie den Fehler.