Fachhochschule Wedel
Prof. Dr. Uwe Schmidt

5. August 1996

Aufgaben zur Klausur **Objektorientierte Programmierung** im SS 96 (IA42)

Zeit: 60 Minuten erlaubte Hilfsmittel: keine

Bitte tragen Sie Ihre Antworten und fertigen Lösungen ausschließlich an den gekennzeichneten Stellen in das Aufgabenblatt ein. Ist ihre Lösung wesentlich umfangreicher, so überprüfen Sie bitte nochmals Ihren Lösungsweg.

Viel Erfolg !

Diese Klausur besteht einschließlich dieses Deckblattes aus 7 Seiten

---

Aufgabe 1:

Gegeben sei das folgende C++–Programmstück, in dem eine Klasse *Str* definiert wird. Dieses Programmstück stehe in der Datei *str0.cc*.

```
#include <string.h>
#include <iostream.h>

class Str {

protected:
  char * sptr;

public:
  Str() {
    sptr = "";
  }

  Str(char * s) {
    sptr = s;
  }

  virtual char operator [] (unsigned int i) const {
    return sptr[i];
  }

  virtual unsigned int len() const {
    return strlen(sptr);
  }

  friend ostream & operator << (ostream & o, const Str & s) {
    return o << s.sptr;
  }

  virtual ~Str() { }
};
```

Hier wird ein *string* in einer *member*–Variablen *sptr* gespeichert, es gibt 2 Konstruktoren, einen Destruktor, die Operatorfunktion zum indizierten Zugriff auf ein Zeichen und eine Funktion zur Berechnung der Länge der gespeicherten Zeichenreihe.

Aus dieser Klasse soll eine Klasse *Str1* entwickelt werden, bei der für die gespeicherten Zeichenreihen beim Konstruktoraufruf immer eine Kopie auf der Halde angelegt wird und beim Destruktoraufruf der Speicher für diese Kopie wieder freigegeben wird. Zusätzlich müssen für diese Klasse dann auch der *copy*–Konstruktor und der Zuweisungsoperator explizit implementiert werden.

Die Klasse hat also folgende Gestalt:

```
#include "str0.cc"

class Str1 : public Str {

public:
  Str1();

  Str1(char * s);

  Str1(const Str1 & s);

  Str1 & operator = (const Str1 & s);

  ~Str1();

private:
  void def_sptr(char * s) {
    sptr = new char[strlen(s) + 1];
    if (! sptr)
      exit(1);
    strcpy(sptr, s);
  }

  void undef_sptr() {
    delete [] sptr;
  }
};
```

Zwei Hilfsfunktionen sind hier schon vorgegeben, implementieren Sie die anderen Funktionen, nur die Funktionsrümpfe, die *header* sind vorgegeben. Diese Klasse soll in der Datei *str1.cc* gespeichert sein.

(Lösungen bitte auf der folgenden Seite)

Str1::Str1()

. . . . . .

Str1::Str1(**char** ∗ s)

. . . . . .

. . . . . .

Str1::Str1(**const** Str1 & s)

. . . . . .

. . . . . .

Str1 & Str1::**operator** = (**const** Str1 & s)

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

Str1::∼Str1()

. . . . . .

Die Klasse *Str1* soll nochmal erweitert werden zu einer Klasse *Str2*, und zwar so, daß die Längenberechnung nur einmal bei der Konstruktion eines Objekts gemacht wird, und nicht bei jedem Aufruf der Methode *len*. Außerdem soll die Operatorfunktion für den indizierten Zugriff sicher gemacht werden, d.h. bei einem illegalen Index soll das Programm abgebrochen werden.

Die Klasse *Str2* hat also folgende Gestalt:

```
#include "str1.cc"

class Str2 : public Str1 {

protected:
  unsigned int slen;

public:
  Str2();

  Str2(char * s);

  Str2(const Str2 & s);

  Str2 & operator = (const Str2 & s);

  ~Str2();

  char operator [] (unsigned int i) const;

  unsigned int len() const;
};
```

Geben Sie die Funktionsrümpfe der *member*–Funktionen an (auf den folgenden Seiten):

Str2::Str2()

. . . . . .

. . . . . .

Str2::Str2(**char** * s)

. . . . . .

. . . . . .

. . . . . .

Str2::Str2(**const** Str2 & s)

. . . . . .

. . . . . .

. . . . . .

Str2 & Str2::**operator** = (**const** Str2 & s)

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

Str2::~Str2()

. . . . . .

char Str2::**operator** [] (**unsigned int** i) **const**

. . . . . .

. . . . . .

. . . . . .

. . . . . .

unsigned Str2::**int** len() **const**

. . . . . .

. . . . . .

. . . . . .

. . . . . .

In der Klasse *Str* sind der indizierte Zugriff und die Längenfunktion als virtuelle Funktionen deklariert. Warum?

Betrachten Sie dazu das folgende Programmstück:

Str & x = Str2("abc");

... x.len() ...
... x[3] ...

. . . . . .

. . . . . .

. . . . . .