Klausurlösung zu OO Softwareentwicklung in C++

2. Februar 2002 (WS 2001/2002)

Bearbeitungszeit: 120 Minuten

NICHT MIT BLEISTIFT SCHREIBEN!!!

Name:	
Vorname:	
Matrikelnummer:	

Aufgabe	Punkte	Bewertung
1	**20**	
2	**10**	
3	**10**	
4	**10**	
(a)	3	
(b)	7	
5	**5**	
6	**6**	
(a)	3	
(b)	3	
7	**12**	
(a)	4	
(b)	2	
(c)	2	
(d)	2	
(e)	2	
8	**7**	
Summe:	**80**	

Bitte benutzen Sie für die Lösungen den freigelassenen Platz nach der jeweiligen Angabe oder die Rückseite unter Angabe der Aufgabe. Nennen Sie möglichst alle Annahmen, die Sie für die Lösung einer Aufgabe treffen!

Vollen Erfolg!!!

Aufgabe 1 **(je 2 Punkte pro Frage = 20 Punkte)**

Bitte jeweils **eine** Antwort ankreuzen (je nach Fragestellung die **richtige** oder **falsche**):

(1) In welchem Zusammenhang taucht das Konzept der virtuellen Klassen auf:
A) Mehrfachvererbung
B) Polymorphie
C) virtuelle Funktionen

(2) Wenn Sie beabsichtigen, eine Klasse A als Basisklasse für andere Klassen einzusetzen, dann gilt:
A) alle Elemente von A sollten protected sein.
B) der Destruktor sollte als virtual deklariert sein.
C) die abgeleitete Klasse muss einen eigenen Konstruktor besitzen.

(3) Der Begriff „virtuelle Methode“ gehört am ehesten zu welchem der folgenden Aspekte?
A) Kapselung
B) Polymorphie
C) Überladen

(4) Was ist hier FALSCH:
A) Jede Mehrfachvererbung kann mit einer HAS-A und einer IS-A-Beziehung simuliert werden.
B) Pointer und Referenzen sind ein und dasselbe.
C) QT hab ich nicht vollkommen durchschaut.

(5) Sie starten den QT-Designers auf der Thales. Was müsse Sie dazu NICHT eingegeben:
A) `export  LD_LIBRARY_PATH=/usr/local/qt-2.3.1/lib`
B) `export QTPATH=/usr/local/qt-2.3.1/bin`
C) `$QTPATH/designer`

(6) Wie heisst die **unmittelbare** Basisklasse, von der QWidget abgeleitet ist?
A) QApplication
B) QT
C) QObject

(7) Was ist hier FALSCH?
Falls benutzerdefinierte Slots verwendet werden,
A) wird der Meta Object Compiler von QT benötigt.
B) wird der User Interface Compiler von QT benötigt.
C) wird der Gnu C++ Compiler benötigt.

(8) Betrachten wir die folgende Codezeile:
`QObject::connect(&quit,SIGNAL(clicked()),&a,SLOT(quit()))`
Welche Antwort wird hier von den Korrekteuren der Klausur am meisten erwünscht:
A) diese Code-Zeile hab ich noch nie gesehen ...
B) halt - ich erinnere mich: Beim Verlassen des Programms, muss vorher unbedingt der Button `quit` gedrückt werden
C) A ist eine Ausrede und B ist - so wie es dasteht - falsch

(9) Welche Klasse ist NICHT von QWidget abgeleitet:
A) QPushButton
B) QLabel
C) QHBoxLayout

(10) Wann setzen Sie ein QGridLayout ein:
A) Wenn leider ein Resize-Event zu erwarten ist.
B) Wenn der Tutor grad in der Nähe ist.
C) Wenn der Prozess mit einem Core-Dump bei einem QBoxLayout abschmiert, und ich nicht weiss, wie ich das beheben soll.

Aufgabe 2 **(10 Punkte)**

Schreiben Sie ein Qt-Programm, das ein Fenster erzeugt, welches dem der linken Abbildung entspricht (Widgets samt Positionierungen sind zu beachten!). Auf der rechten (grossen) Abbildung können Sie ablesen, wie das Fenster nach einem Resize-Event aussehen wird. Achten Sie bitte auf den oberen und unteren Leeraum zwischen dem Fenster und den Widgets.

Die Erzeugung dieses Fensters kann in `main()` erfolgen. Sie müssen keine eigene Qt-Widget-Klasse implementieren, es reicht die Verwendung bekannter Widgets im Hauptprogramm. Ausserdem können Sie auf die Angabe der Include-Dateien verzichten.

Lösung:

```
#include <qapplication.h>
#include <qwidget.h>
#include <qlayout.h>
#include <qlabel.h>
#include <qcheckbox.h>
#include <qpushbutton.h>

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

    // Widget mit Gitterlayout (am Rand 10 Punkt Freiraum)
    QWidget widget;
    // ... und alle enthaltenen Widgets in einem Gitter anordnen
    QGridLayout *l = new QGridLayout(&widget, 1, 1, 10);

    // erste und letzte Zeile sollen nur "wachsen"
    // genauso die erste Spalte
    l->setRowStretch(0, 1);
```

```
    l->setRowStretch(4, 1);
    l->setColStretch(0, 1);

    // links kommen in die Mitte 3 Checkboxes
    QCheckBox *c1 = new QCheckBox("Apfel", &widget);
    l->addWidget(c1, 1, 0);
    QCheckBox *c2 = new QCheckBox("Birne", &widget);
    l->addWidget(c2, 2, 0);
    QCheckBox *c3 = new QCheckBox("Banane", &widget);
    l->addWidget(c3, 3, 0);

    // rechts stehen in der Mitte 3 Buttons
    QPushButton *b1 = new QPushButton("Fertig", &widget);
    l->addWidget(b1, 1, 1);
    QPushButton *b2 = new QPushButton("Abbrechen", &widget);
    l->addWidget(b2, 2, 1);
    QPushButton *b3 = new QPushButton("Zusammenbrechen", &widget);
    l->addWidget(b3, 3, 1);

    // Hauptwidget setzen ... und anzeigen
    app.setMainWidget(&widget);
    widget.show();

    return app.exec();
}
```

Aufgabe 3 **(10 Punkte)**

Implementieren Sie eine Klasse `Translator`, die sowohl einzelne Worte, als auch eine Folge von Worten, welche durch genau ein Zeichen getrennt sind, übersetzen kann. Ein passendes Wörterbuch liegt bereits in der Datei `dic.dat` vor.
Jeweiliger Aufbau der Zeilen: `Wort:Uebersetzung`
Links steht das zu übersetzende Wort und rechts vom `:` die passende Übersetzung.
Hier ein Auszug aus der Datei `dic.dat`:

```
Hallo:Hello
Welt:World
```

Aus einem bereits geöffneten und übergebenen Input-Stream liest `Translator` im Konstruktor das Wörterbuch in der obigen Form ein (siehe unten im Main-Programm).
Die Klasse `Translator` muss mindestens die folgenden beiden öffentlichen Methoden implementieren:

```
string translate(string word);
void translate(istream &is, ostream &os, char delim=' ');
```

Die 1.Methode `translate` liefert die Übersetzung des übergebeben Wortes als String zurück. Falls das übergebene Wort nicht im Wörterbuch enthalten ist, wird einfach das Originalwort in spitzen Klammern statt der Übersetzung zurückgeliefert (siehe ganz unten im Beispiel).
Das zweite `translate` agiert analog: es übersetzt sämtliche mit `delim`-getrennten Wörter des Eingabestroms und schreibt das Ergebnis in den Ausgabestrom.
Es soll das folgende Hauptprogramm mit Ihrer Klasse `Translator` lauffähig sein:

```
void main() {
    ifstream f("dic.dat");
    Translator t(f);

    t.translate(cin, cout);
}
```

Beispiel für eine Ausführung dieses Hauptprogramms mit dem oben angegebenen Wörterbuch:

```
thales$ cat text
Dies ist Hallo Welt
thales$ g++ dict.cc
thales$ a.out < text
<Dies> <ist> Hello World
thales$
```

Lösung:

```
#include <iostream>
#include <fstream>
#include <string>
#include <map>

class Translator {
    public:

    Translator(istream &is) : dic() {
        read(is);
    }

    string translate(string word);
    void translate(istream &is, ostream &os, char delim=' ');

    private:

    map<string,string> dic;
    void read(istream &is);
};

string Translator::translate(string word) {
    if (dic.find(word) != dic.end())
        return dic[word];
    else
        return string("<").append(word).append(">");
}

void Translator::translate(istream &is, ostream &os, char de-
lim) {
    char word[200];
    while (true) {
        is.getline(word, 200, delim);
        if (!is) break;
        os << translate(word) << delim;
    }
}

void Translator::read(istream &is) {
    char word[200], trans[200];
    while (true) {
```

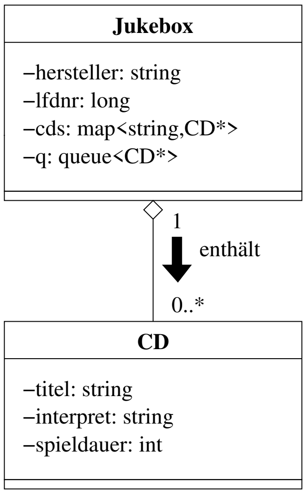
```
        is.getline(word, 200, ':').getline(trans, 200, '\n');
        if (!is) break;
        dic[word] = trans;
    }
}

void main() {
    ifstream f("dic.dat");
    Translator t(f);

    t.translate(cin, cout);
}
```

Aufgabe 4 **(10 Punkte)**

Eine Jukebox ist eine Sammlung von vielen CDs. Sie besitzt eine laufende Seriennummer. Die Jukebox kann von aussen nach und nach mit beliebig vielen CDs geladen werden. Zum Start der Box können immer 3 CDs aus der Sammlung durch Angabe des Titels der CD ausgewählt werden. Danach werden sie in der Reihenfolge der Auswahl abgespielt. Nach Auswahl der dritten (korrekt angegebenen) CD faengt die Jukebox automatisch zu spielen an, indem Sie den Titel der CD und den Interpreten auf `cout` ausgibt `:-))`
Eine CD besteht aus einem Titel, dem Namen des Interpreten und der Gesamtlänge aller abzuspielenden Songs in Minuten.

(a) 3 Punkte
Zeichnen Sie ein UML-Diagramm von den benötigten Klassen unter Angabe der Assoziationen und Vererbungsrichtungen zwischen den Klassen sowie der benötigten Klassenattribute. Auf die Angabe der Elementfunktionen können Sie der Einfachheit halber verzichten.

Lösung:



(b) 7 Punkte
Schreiben Sie eine vollständige C++-Klassendefinition (samt Implementierung der Klassenmethoden!), die eine CD-Jukebox abbildet. Die Include-Anweisungen können sie der Einfachheit halber weglassen.
Das main-Programm finden Sie als Orientierungshilfe hier angegeben:

```
main() {
  Jukebox jui("Grammy", 1235566);
```

```
  jui.loadCD(new CD("Requiem","Wolferl Mozart", 12));
  jui.loadCD(new CD("Diesel Vanilla","Ezio", 24));
  jui.loadCD(new CD("Both Sides","Phil Collins", 26));
  jui.selectCD("Diesel Vanilla");
  jui.selectCD("Hard’n Heavvy");
  jui.selectCD("Requiem");
  jui.selectCD("Diesel Vanilla");
}
```

Lösung:

```
#include <string>
#include <vector>
#include <queue>
#include <iostream>
#include <map>
class CD {
        string label, interpret;
        int spieldauer;
public:
        CD(string l="", string name="", int dauer=0):
                        label(l),interpret(name),spieldauer(dauer) {}
        string getLabel() const {       return label; }
        void showCD() { cout <<interpret<<": "<<label<<endl; }
};
class Jukebox {
        map<string, CD*> cds;
        queue<CD*> q;
        string date;
        long lfdnr;
public:
        Jukebox(string herstell, long lfd): date(herstell), lfd-
nr(lfd) {}
        void loadCD(CD *cd) {   cds[cd->getLabel()] = cd;       }
        void abspielen()
        {
                while (!q.empty())
                        (q.front())->showCD(),
                        q.pop();
        }
        void selectCD(string label)
        {
                CD *cd = cds[label];
```

```
                if (cd) {
                        q.push(cd);
                        if (q.size()==3)
                                abspielen();
                }
        }
};

main()
{
        Jukebox jui("Grammy", 1235566);

        jui.loadCD(new CD("Requiem","Wolferl Mozart", 12));
        jui.loadCD(new CD("Diesel Vanilla","Ezio", 24));
        jui.loadCD(new CD("Both Sides","Phil Collins", 26));
        jui.selectCD("Diesel Vanilla");
        jui.selectCD("Hard’n Heavvy");
        jui.selectCD("Requiem");
        jui.selectCD("Diesel Vanilla");
}
```

Aufgabe 5 **(5 Punkte)**

Welche Ausgabe erzeugt das folgende Programm?

```
#include <iostream>

int g = 0;

void obscure(int &a, int &b = g) {
    if (a > b) { int tmp = a; a = b; b = tmp; }
    else { int tmp = a; a = g; g = b; }
}

void obscure(int *a) {
    int tmp = *a; *a = g; g = tmp;
}

void obscure(const int *a) {
    g = *a;
}

template <class T>
T &obcure(T &a) {
    return a;
}

void main() {
    int x = 2;
    cout << "x = " << x << ", g = " << g << endl;
    obscure(x);
    cout << "x = " << x << ", g = " << g << endl;
    obscure(g, x);
    cout << "x = " << x << ", g = " << g << endl;
    obcure(x) = 7;
    cout << "x = " << x << ", g = " << g << endl;
    obscure(&x);
    cout << "x = " << x << ", g = " << g << endl;
}
```

Lösung:

```
thales$ try obscure.cc
x = 2, g = 0
x = 0, g = 2
x = 2, g = 0
x = 7, g = 0
x = 0, g = 7
```

Aufgabe 6 **(6 Punkte)**

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

class A {
protected:    float *koord;
public:
    static int cols;
    A(float *init = 0) {
        koord = new float[cols];
        if (init)
            memcpy(koord, init, cols*sizeof(float));
        else
            memset(koord, 0, cols*sizeof(float));
    }
    virtual ~A() { *koord = 0; }
};
class B : public A {
public:
    B(float *init=0):A(init) { }
    void dispKoord() {
        for (int i=0; i<A::cols; i++)
            cout <<koord[i] <<" ";
        cout <<endl;
    }
};
int A::cols = 0;
void main() {
    float fa[] = {1, 2, 3, 4};
    B::cols = sizeof(fa)/sizeof(*fa);
    B b;
    {
        B b4(fa);
        b = b4;
    }
    b.dispKoord();
}
```

(a) (3 Punkte) Was gibt dieses Programm aus?

Lösung:

```
thales$ try murks.cc
0 2 3 4
```

(b) (3 Punkte) Wie kann durch Änderung der Klassendefinition (nicht des Hauptprogramms!) die wohl eigentlich gewünschte Ausgabe erreicht werden?

Lösung:

```
class B : public A {
public:
        B(float *init=0):A(init) { }
        void dispKoord() {
                for (int i=0; i<A::cols; i++)
                        cout <<koord[i] <<" ";
                cout <<endl;
        }
#ifdef LOESUNG
        B &operator=(B const &b)
        {
                koord = new float[cols];
                memcpy(koord, b.koord, cols*sizeof(float));
                return *this;
        }
#endif
};
```

Aufgabe 7 **(12 Punkte)**

Gegeben ist das folgende C++-Programm (in Kommentaren sind Änderungen angegeben, die für bei die Teilaufgaben (b)-(e) notwendig sind):

```
#include <iostream>

class Mensch {
    public:
    Mensch() {
        cout << "Ein Mensch wird geboren." << endl;
    }
    /* virtual */ ~Mensch() {
        cout << "Ein Mensch stirbt." << endl;
    }
    /* virtual */ void sagWas() {
        cout << "Hallo Freunde!" << endl;
    }
    virtual void woherBistDu() {
        cout << "Ich lebe auf der Erde." << endl;
    }
};

class Italiener : public /* virtual */ Mensch {
    public:
    Italiener() {
        cout << "Ein Italiano wird geboren." << endl;
    }
    ~Italiener() {
        cout << "Italiano stirbt." << endl;
    }
    /* virtual */ void sagWas() {
        cout << "Ciao amici!" << endl;
    }
    void woherBistDu() {
        cout << "Isch komme aus Italia." << endl;
    }
};

void main() {
    Mensch *m = new Italiener();
    m->woherBistDu();
    m->sagWas();
    delete m;
}
```

Beantworten Sie nun folgende Fragen zu diesem Programm jeweils **unter Angabe einer kurzen Begründung**.

(a) (4 Punkte) Welche Ausgabe produziert dieses Programm?

Lösung:

```
Ein Mensch wird geboren.
Ein Italiano wird geboren.
Isch komme aus Italia.
Hallo Freunde!
Ein Mensch stirbt.
```

Implizit wird außer dem Konstruktor von `Italiener` auch der Default-Konstruktor von `Mensch` aufgerufen, wenn kein Aufruf eines Oberklassen-Konstruktors angegeben ist. Weil `woherBistDu()` virtuell ist, wird die richtige Methode von Italiener aufgerufen. Dahingegen ist `sagWas()` nicht virtuell und somit wird die Methode von `Mensch` aufgerufen. Weil schließlich der Destruktor von `Mensch` nicht virtuell ist, wird der Destruktor von `Italiener` nicht vor dem von `Mensch` aufgerufen.

(b) (2 Punkte) Wie ändert sich die Ausgabe, wenn man den Destruktor in Mensch als “virtual“ deklariert?

Lösung:

Vor der Zeile `Ein Mensch stirbt.` kommt nun die Zeile `Italiano stirbt.` hinzu, weil der Destruktor von `Mensch` nun virtuell ist und somit zur Laufzeit noch der richtige Konstruktor von `Italiener` noch *zusätzlich* aufgerufen wird.

(c) (2 Punkte) Was ändert sich an der Ausgabe, wenn die Methode sagWas() in Italiener als “virtual“ deklariert wird?

Lösung:

Da ändert sich gar nix, da das virtual nicht an die Basisklasse mitgeteilt wird. Also wird sagWas der Basisklasse aufgerufen!

(d) (2 Punkte) Was ändert sich an der Ausgabe, wenn die Methode sagWas() in Mensch als “virtual“ deklariert wird?

Lösung:

Es wird nun statt `Hallo Freunde!` der Text `Ciao amici!` ausgegeben, da bei einer virtuellen Methode die richtige Implementierung in der Unterklasse gefunden wird (zur Laufzeit – dynamische Bindung!).

(e) (2 Punkte) Inwiefern ändert sich die Ausgabe des Programms, wenn Mensch zu einer virtuellen Basisklasse gemacht wird, d.h. Italiener virtuell von Mensch erbt? Wann ist virtuelles Erben erforderlich?

Lösung:

Da ändert sich nix mehr ... virtuelle Klassen nur bei Mehrfachvererbung nötig ... Mehrfachvererbung liegt hier nicht vor.

Aufgabe 8 **(7 Punkte)**

Was gibt das Programm aus?
Lösung: `objcnt is 3`

Kennzeichnen Sie die Stellen im Programm, an denen neue Objekte der Klasse `oih` angelegt werden.

```
#include <iostream>
class oih {
public:
        static int objcnt;
        oih() { objcnt++; }
        oih(oih const &uk) { objcnt++; }
        oih &operator=(oih const &uk) { objcnt++; }
};
int oih::objcnt=0;
void uih(oih ahh) { /* NEUES OIH bei Parameteruebergabe */
        oih quik = ahh; /* NEUES OIH als lokale Variable */
}
main() {
        oih *a = new oih(); /* NEUES OIH */
        oih *b = a;
        uih(*b);
        cout <<"objcnt is "<<oih::objcnt<<endl;
}
```

Lösung: Kennzeichung durch Kommentare im Programm