

Sommersemester 2008	Zahl der Blätter: 15 Blatt Nr: 1
Fachbereich: Informationstechnik	Semester: IT3A/B
Prüfungsfach: Informatik 3	Fachnummer: KTB/SWB/TIB 3011
Hilfsmittel: keine elektronischen Hilfsmittel	Zeit: 90 min
Name:	Matrikel-Nr.:

Hinweis: Der auf den Blättern jeweils freigelassene Raum reicht im Allgemeinen vollständig für die stichwortartige Beantwortung der Fragen, bzw. für die Lösungen aus. Tragen Sie daher auf **jedem** Blatt Ihren Namen und Ihre Matrikelnummer ein und nutzen Sie diese Blätter zur Abgabe Ihrer Antworten und Lösungen.

Aufgabe 1: Allgemeine Fragen

(ca. 20 Min.)

Bitte beurteilen Sie die folgenden allgemeinen Aussagen. Machen Sie jeweils ein Kreuzchen in der Spalte „wahr“ oder „falsch“. Begründen Sie jeweils Ihre Wahl.

Aussage	wahr	falsch
<p>1) Gegeben sei folgende Klasse <code>class Konto {};</code></p> <p>Dann liefert folgende Zeile einen Compilerfehler: <code>Konto& k = *(new Konto);</code></p> <p>Begründung:</p>		
<p>2) Konstante Instanzvariablen einer Klasse können unterschiedlich für mehrere Instanzen dieser Klasse sein.</p> <p>Begründung:</p>		

Sommersemester 2008	Blatt Nr: 2 / 15
Prüfungsfach: Informatik 3	Fachnummer: KTB/SWB/TIB 3011
Name:	Matrikel-Nr.:

Aussage	wahr	falsch
<p>3) Eine nicht statische Methode der Klasse A kann auf private statische Klassenvariablen von A zugreifen. Begründung:</p>		
<p>4) Mit</p> <pre>void swap(int a, int b) {int t=a; a=b; b=t;}</pre> <p>können die Werte von zwei Variablen ausgetauscht werden. Begründung:</p>		
<p>5) Die Basisklasse enthalte eine rein virtuelle Methode. Von einer davon abgeleiteten Klasse können immer Instanzen gebildet werden. Begründung:</p>		
<p>6) Virtuelle Konstruktoren können in abgeleiteten Klassen überladen werden. Begründung:</p>		

Sommersemester 2008	Blatt Nr:	3 / 15
Prüfungsfach: Informatik 3	Fachnummer:	KTB/SWB/TIB 3011
Name:	Matrikel-Nr.:	

<p>7) Gegeben sei eine Klasse D; dann kann eine Klasse A folgendermaßen definiert und verwendet werden</p> <pre>class D {}; class A { const D data; }; void main() { A a; }</pre> <p>Begründung:</p>		
<p>8) Gegeben sei eine Klasse D, die als Komposition ein Teilobjekt vom Datentyp K enthält; der Konstruktor von D wird zuletzt abgearbeitet.</p> <p>Begründung:</p>		
<p>9) Ein unärer Operator ++ wird als Methode einer Klasse überladen. Die Parameterliste der zugehörigen Operatorfunktion ist immer leer.</p> <p>Begründung:</p>		

Sommersemester 2008	Blatt Nr:	4 / 15
Prüfungsfach: Informatik 3	Fachnummer:	KTB/SWB/TIB 3011
Name:	Matrikel-Nr.:	

<p>10) Gegeben sei eine Klasse Konto sowie zwei globale Methoden:</p> <pre>void print(Konto& k) {}; void print(const Konto& k, int i=0) {};</pre> <p>Der Compiler erzeugt keine Fehlermeldung bei folgenden Aufrufen:</p> <pre>Konto k; print(k);</pre> <p>Begründung:</p>		
---	--	--

Sommersemester 2008	Blatt Nr: 5 / 15
Prüfungsfach: Informatik 3	Fachnummer: KTB/SWB/TIB 3011
Name:	Matrikel-Nr.:

Aufgabe 2: Klassendefinition

(ca. 15 Min.)

Sie sollen einen Automaten programmieren, an dem man Schokoriegel kaufen kann.

Die Klasse `schokoautomat` soll folgende Eigenschaften besitzen:

- a) Instanzvariablen `anz_riegel` vom Typ `int`, die die Anzahl im Automaten verbleibenden Schokoriegel speichert
- b) eine Instanzvariable `summe` vom Typ `int`, die mitzählt wie viel Cent für den Kauf eines Schokoriegels eingeworfen wurde
- c) eine Instanzvariable `umsatz` vom Typ `int`. Diese Variable gibt an, wie viel Geld (in Cent) bisher vom Automaten eingenommen wurde.
- d) eine konstante Instanzvariable `nr` vom Typ `int`, die die Seriennummer des Automaten speichert. Die Seriennummer wird automatisch in Abhängigkeit einer Klassenvariablen `maxnr` vergeben.
- e) eine Klassenvariable `preis`, die den Preis in Cent eines Schokoriegels für alle Schokoautomaten speichert. Initialisieren Sie den Preis mit 0.
- f) ein Konstruktor, der die Instanzvariablen eines Automaten sinnvoll initialisiert. Die Instanzvariable `anz_riegel` soll dabei über einen Defaultparameter auf 50 gesetzt werden.
- g) ein Destruktor, der `anz_riegel` und `umsatz` auf 0 setzt.
- h) eine Instanzmethode `riegel_ausgeben()`, die die Variable `anz_riegel` entsprechend ändert und den Umsatz erhöht, wenn die Variable `summe` den Riegelpreis erreicht hat
- i) eine Instanzmethode `muenze_einwerfen(...)`, die, wenn sich noch Schokoriegel im Automaten befinden, die Variable `summe` um den eingeworfenen Centbetrag erhöht. Die Münzen werden über den Aufzählungstyp `muenzen` repräsentiert. Diese Instanzmethode ruft automatisch die Methode `riegel_ausgeben()` auf, wenn der Riegelpreis erreicht wurde.
- j) eine Methode `getUmsatz()`, die den Umsatz in Euro (double) zurückliefert
- k) eine Klassenmethode `setPreis(...)`, die den Riegelpreis in Cent setzt
- l) ein kaskadierbaren Operator `+`, der es erlaubt, `a.muenze_einwerfen(cent50)` über `a+cent50` aufzurufen
- m) ein Ausgabeoperator `<<`, der mit einem Schokoautomaten als Operanden die Nummer, die Anzahl der verbleibenden Riegel sowie den Umsatz an den Ausgabestream übergibt

Sommersemester 2008	Blatt Nr: 6 / 15
Prüfungsfach: Informatik 3	Fachnummer: KTB/SWB/TIB 3011
Name:	Matrikel-Nr.:

Ergänzen Sie das folgende Programmgerippe. Schützen Sie die Datenelemente vor Zugriffen durch klassenfremde Methoden.

Prototypen der Klasse (Aufgabe 2)

```
#include
```

```
enum muenzen { cent10=10, cent20=20, cent50=50, euro1=100,  
              euro2=200 };
```

```
class Schokoautomat  
{
```

```
public:
```

```
    // Prototyp des Konstruktors
```

```
    // Prototyp des Destruktors
```

```
    // Prototyp von muenze_einwerfen
```

```
    // Prototyp von riegel_ausgeben
```

```
    // Prototyp von getUmsatz
```

```
    // Prototyp von setPreis
```

```
    // Prototyp von Operator +
```

Sommersemester 2008	Blatt Nr: 7 / 15
Prüfungsfach: Informatik 3	Fachnummer: KTB/SWB/TIB 3011
Name:	Matrikel-Nr.:

`// Prototyp vom Ausgabeoperator`

`};`

Aufgabe 3: Implementierung der Methoden und Operatoren der Klasse (ca. 20 Min)

Programmieren Sie hier und auf den folgenden Seiten außerhalb der Klasse `Schokoautomat` die angegebenen Elemente aus.

Schreiben Sie dann eine Hauptfunktion, die

- den Riegelpreis auf 1,50 Euro setzt
- einen Schokoautomaten `s1` mit 100 Riegeln konstruiert
- 1,50 Euro in `s1` einwirft
- `s1` ausgibt

`// Konstruktor`

`// Destruktor`

Sommersemester 2008	Blatt Nr:	8 / 15
Prüfungsfach: Informatik 3	Fachnummer:	KTB/SWB/TIB 3011
Name:	Matrikel-Nr.:	

```
// Methode muenze_einwerfen
```

```
// Methode riegel_ausgeben
```

```
// Methode getUmsatz
```

```
// Methode setPreis
```

Sommersemester 2008	Blatt Nr:	9 / 15
Prüfungsfach: Informatik 3	Fachnummer:	KTB/SWB/TIB 3011
Name:	Matrikel-Nr.:	

```
// Operator +
```

```
// Operator <<
```

```
// Klassenvariablen initialisieren
```

```
int main()  
{
```

```
}
```

Sommersemester 2008	Blatt Nr:	10 / 15
Prüfungsfach: Informatik 3	Fachnummer:	KTB/SWB/TIB 3011
Name:	Matrikel-Nr.:	

Aufgabe 4: Polymorphie

(ca. 10 min)

Analysieren Sie das nachfolgende Programm und schreiben Sie die Ausgabe des Programms auf.

```
#include <iostream>
#include <string>
using namespace std;

class Form {
protected:
    string name;
public:
    Form(string n = "none"):name(n) {
        cout << endl << " b " << *this;}
    friend ostream& operator<<(ostream& os, const Form& f);
    virtual void draw() const {cout << *this; }
};

class Polygon : public Form {
protected:
    int ecken;
public:
    Polygon(string n = "Polygon", int e=4):
        Form(n), ecken(e) {}
    friend ostream& operator<<(ostream& os, const Polygon& p);
    void draw() const { cout << " p " << *this; }
};

class Triangle : public Polygon {
public:
    Triangle():Polygon("Tri", 3) { cout << " T "; }
    void draw() const { cout << " ^ " << *this; }
};

class Rectangle : public Polygon {
public:
    Rectangle():Polygon("Rect", 4) { cout << " R "; }
    void draw() const {cout << " # " << *this ; }
};

ostream& operator<<(ostream& os, const Form& f) {
    cout << "F " << f.name; return os;
}
```

Sommersemester 2008	Blatt Nr:	11 / 15
Prüfungsfach: Informatik 3	Fachnummer:	KTB/SWB/TIB 3011
Name:	Matrikel-Nr.:	

```
ostream& operator<<(ostream& os, const Polygon& p) {
    cout << "P " << p.name; return os;
}

int main()
{
    Form* f[] = {
        new Polygon("Test", 5),
        new Rectangle(),
        new Triangle()
    };
    cout << endl << endl;
    for (int i=0; i<3; ++i) {
        f[i]->draw();
        cout << endl;
    }
    Form& g = (*f[2]);
    g.draw();
    cout << endl;
    const Triangle t;
    t.draw();
    cout << endl;
    return 0;
}
```

Ausgabe des Programms:

Sommersemester 2008	Blatt Nr:	12 / 15
Prüfungsfach: Informatik 3	Fachnummer:	KTB/SWB/TIB 3011
Name:	Matrikel-Nr.:	

Aufgabe 5: Vererbung

(ca. 10 Min.)

Schauen Sie sich das nachfolgende Programm an und schreiben Sie dessen Ausgabe nieder.

```
#include <iostream>
using namespace std;

class A
{
public:
    int a;
    A(int a=0) : a(a) {}
};

class B : virtual public A
{
public:
    int b;
    B(int b=0) : b(b) {}
    B(int a, int b) : A(a), b(b) {}
};

class C1 : public B
{
public:
    int c1;
    C1(int c1=0) : c1(c1) {}
    C1(int a, int b, int c1) : B(a+10, b+10), c1(c1+10) {}
};

class C2 : public B
{
public:
    int c2;
    C2(int c2=0) : c2(c2) {}
    C2(int a, int b, int c2) : B(a+20, b+20), c2(c2+20) {}
};
```

Sommersemester 2008	Blatt Nr:	13 / 15
Prüfungsfach: Informatik 3	Fachnummer:	KTB/SWB/TIB 3011
Name:	Matrikel-Nr.:	

```
class D : virtual public C1, virtual public C2
{
public:
    int d;
    D(int a, int b, int c1, int c2, int d)
        : C1(a, b, c1), C2(a, b, c2), d(d) {}
};

int main()
{
    D obj(1, 2, 3, 4, 5);
    cout << "a = " << obj.C1::a << endl;
    cout << "a = " << obj.C2::a << endl;
    cout << "b = " << obj.C1::b << endl;
    cout << "b = " << obj.C2::b << endl;
    cout << "c1 = " << obj.c1 << endl;
    cout << "c2 = " << obj.c2 << endl;
    cout << "d = " << obj.d << endl;
    cout << "Groesse von obj: " << sizeof(obj) << endl;
    return 0;
}
```

Ausgabe:

Sommersemester 2008	Blatt Nr:	14 / 15
Prüfungsfach: Informatik 3	Fachnummer:	KTB/SWB/TIB 3011
Name:	Matrikel-Nr.:	

Aufgabe 6: Templates

(ca. 15 Min.)

Ergänzen Sie das folgende Programmgerüst, so dass das Programm fehlerfrei ausgeführt werden kann und die gegebene Ausgabe erzeugt. Die Größe des Stacks wird an das Template übergeben und soll bei Push- und Pop-Operationen überprüft werden, die das Ergebnis der Prüfung als `bool` zurückgeben.

```
#include <iostream>
#include <string>
using namespace std;
```

```
class Stack {
private:
```

```
public:
    Stack()
```

```
};
```

Sommersemester 2008	Blatt Nr:	15 / 15
Prüfungsfach: Informatik 3	Fachnummer:	KTB/SWB/TIB 3011
Name:	Matrikel-Nr.:	

```

void main() {
    Stack<int, 10> intstack;
    int i = 0;
    while (!intstack.isfull()) {
        intstack.push(++i);
    }
    while (intstack.pop(i)) {
        cout << i << " ";
    }
    cout << endl;

    string var;    char c = 'a';
    Stack<string, 2> stringstack;
    while (!stringstack.isfull()) {
        var = string("s_") + c++;
        stringstack.push(var);
    }
    while (stringstack.pop(var)) {
        cout << var << " ";
    }
    cout << endl;
}

```

Ausgabe:

```

10 9 8 7 6 5 4 3 2 1
s_b s_a

```