Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim Geisenheim

Fachbereich Design Informatik Medien
Studiengang Bachelor Allgemeine Informatik

3071 Programmieren 3 AI

Klausur

WS 2011/12

2012-02-24

Name:

Vorname:

Matrikelnr.:

Unterschrift

Erlaubte Hilfsmittel: keine.

Die Lösungen sind auf den Klausurbögen anzufertigen. Verwenden Sie einen dokumentenechten Stift nichtroter Farbe. Lösen Sie die Heftung der Klausur nicht. Notieren Sie die Verwendung von Rückseiten auf den entsprechenden Vorderseiten.

Bearbeitungszeit: 90 Minuten.

Aufgabe	1	2	3	4	5	Gesamt	Note
Punkte	16	20	20	24	20	100	–
erreicht							

Aufgabe 1 (16 Punkte)

Das folgende Java-Programm enthält ein Synchronisationsproblem. Erläutern Sie, wodurch das Problem entsteht und wieso der Quelltext in der angegebenen Form ungeeignet ist. Beziehen Sie sich dabei auch auf die Codezeilen, durch die das Problem zustandekommt, möglichst unter Verwendung entsprechender Fachbegriffe. Beschreiben Sie außerdem eine Möglichkeit, das Problem zu beheben.

```
1  class Sequence {
2      volatile private int n = 0;
3      public int next() {
4          return ++n;
5      }
6  }
7
8  class Worker extends Thread {
9      private Sequence s;
10     public Worker(Sequence s) {
11         this.s = s;
12     }
13     public void run() {
14         for (int i = 0; i < 100; i++) {
15             int seqno = s.next();
16         }
17     }
18 }
19
20 class Main {
21     private final static int N_THREADS = 100;
22     public static void main(String[] args)
23         throws InterruptedException {
24         Worker[] w = new Worker[N_THREADS];
25         Sequence s = new Sequence();
26         for (int i = 0; i < N_THREADS; i++) {
27             w[i] = new Worker(s);
28             w[i].start();
29         }
30         for (int i = 0; i < N_THREADS; i++) {
31             w[i].join();
32         }
33         System.out.println("Nächste Nummer: " + s.next());
34     }
35 }
```

Aufgabe 2 (20 Punkte)

Jeder der folgenden C++-Quelltexte enthält einen Fehler, der die Übersetzung zu einem ausführbaren Programm verhindert. Erläutern Sie den Fehler und beschreiben Sie, wie er behoben werden kann.

(a)

```
1  #include <iostream>
2
3  class A {
4  public:
5      std::string getFinalMessage();
6  };
7
8  std::string getFinalMessage() {
9      return "we apologise for the inconvenience";
10 }
11
12 int main() {
13     std::cout << (new A())->getFinalMessage() << std::endl;
14 }
```

(b)

```
1  #include <vector>
2  #include <complex>
3
4  int main() {
5      std::vector<std::complex<double>> cmplvec;
6      cmplvec.push_back(std::complex<double>(0, 1));
7      cmplvec.push_back(std::complex<double>(17, 23));
8      cmplvec.push_back(std::complex<double>(105, -42));
9      cmplvec.push_back(666);
10     return 0;
11 }
```

(c)

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

class Person {
public:
    string name;
    string vorname;

    Person(string name, string vorname);
    string getFullName();
};

Person::Person(string name, string vorname) :
    name(name), vorname(vorname) {}

string Person::getFullName() {
    return vorname+" "+name;
}

int main() {
    char* n = "Zerlett";
    string v = "Helmut";
    Person p = new Person(n, v);
    cout << p.getFullName() << endl;
    cout << p.vorname << endl;
}
```

(d)

```
class A {
public:
    int i;
    A(int i);
};

A::A(int i) : i(i) {}

int main() {
    A a;
    A b = 42;
    return 0;
}
```

(e)

```
int f(int& x) {
    x = x + 2;
    return x;
}

int main() {
    f(23);
    return 0;
}
```

Aufgabe 3 (20 Punkte)

Berechnen Sie die Ausgabe der folgenden C++-Programme von Hand; verzeichnen Sie dabei ggf. auch nicht gefangene Exceptions. Erklären Sie kurz, wie es zu der Ausgabe kommt.

(a)

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

class C1 {
public:
    virtual string getDescription();
};

class C2 : public C1 {
public:
    virtual string getDescription();
};

string C1::getDescription(){
    return "C1";
}

string C2::getDescription(){
    return "C2";
}

int main() {
    C1 c1;
    C2 c2;
    c1 = c2;
    C1* pC1 = &c2;

    cout << c1.getDescription() << endl;
    cout << c2.getDescription() << endl;
    cout << pC1->getDescription() << endl;
    pC1 = new C1();
    cout << pC1->getDescription() << endl;
}
```

(b)

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

template <typename T> class Box {
public:
    T content;
    Box<T>(T x) : content(x) {}
};

int f(int& x) {
    x = x + 4;
    return x;
}

template <typename T> void fB(Box<T> b) {
    b.content = f(b.content);
}

int main(){
    int y = 17;
    int z = (f(y));
    Box<int> b(105);
    fB(b);

    cout << z << endl;
    cout << y << endl;
    cout << b.content << endl;
}
```

(c)

```
#include <iostream>

int f1(int x) {
    int result = 0;
    for (int i = 0; i <= x; i++) {
        if (i == 2) throw result;
        if (x < 0) throw "negatives Argument in f1";
        result = result + i;
    }
    return result;
}

int f2(int i) {
    try {
        return f1(i);
    } catch (int i) {
        return i;
    }
}

int main() {
    std::cout << f2(7) << std::endl;
    return 0;
}
```

(d)

```
#include <iostream>

template <int N> class A {
public:
    enum { v = N * A<N-1>::v };
};

template <> class A<0> {
public:
    enum { v = 1 };
};

int main() {
    std::cout << A<4>::v << std::endl;
    return 0;
}
```

Aufgabe 4 (24 Punkte)

(a) Schreiben Sie folgende Funktion im Stil der Algorithmen der Standardbibliothek:

```
1 template <typename Iterator, typename ElementType>
2 void reverse(Iterator anfang, Iterator ende,
3     std::deque<ElementType>& result);
```

Die Funktion soll alle Elemente des Iterationsbereichs in umgekehrter Reihenfolge in die übergebene Deque einfügen. Achten Sie darauf, dass die Funktion für alle Typen von Iteratoren verwendbar ist.

(b) Gegeben sei folgende Template-Klasse, die einen binären Suchbaum realisiert:

```
1 template <typename T> class TreeNode {
2 private:
3     T wert;
4     TreeNode<T>* leftChild;
5     TreeNode<T>* rightChild;
6 public:
7     TreeNode(T w) : wert(w), leftChild(0), rightChild(0) {}
8 };
```

Jeder Knoten ist Eigentümer seiner Kindknoten, sofern diese existieren. Ergänzen Sie die Klasse um

1. einen geeigneten Destruktor
2. die Methode `void inorder(std::vector<T>& result) const;` die die im Baum gespeicherten Werte in Inorder-Reihenfolge in `result` einfügt.
3. die Methode `void insert(TreeNode<T>* node);` die den übergebenen Knoten korrekt sortiert in den Baum einfügt (ggf. vermittels eines rekursiven Aufrufs); beachten Sie bzgl. der Speicherverwaltung dass durch den Aufruf dieser Methode das Eigentum am einzufügenden Knoten übertragen wird. Falls bereits ein Knoten mit demselbem Wert im Baum existiert soll der neue Knoten nicht in den Baum eingefügt, sondern verworfen werden; achten Sie vor allem bei diesem Fall auf korrekte Speicherverwaltung. Sie können davon ausgehen dass die Vergleichsoperatoren (`<`, `<=`, `>`, `>=`, `==`, `!=`) für den Typ `T` definiert sind.

Aufgabe 5 (20 Punkte)

(a) Wann ist es in C++ nötig, für eine Klasse einen Destruktor explizit zu programmieren?

(b) Was versteht man in C++ unter einem Default-Konstruktor?

(c) Warum sollte in C++ der Datentyp einer `catch`-Klausel immer ein Referenztyp sein?

(d) Wie kann das Schlüsselwort `explicit` in C++ verwendet werden, und was bewirkt es? Wann ist sein Einsatz sinnvoll?

(e) Das Und-Zeichen `&` hat in C++ mehrere Bedeutungen. Welche?

(f) Das Exception-Handling von C++ kennt (im Gegensatz zu Java) keine `finally`-Klausel. Warum wird sie nicht benötigt?

(g) Aus welchen Schritten besteht das Übersetzen eines C++-Quelltexts zu einem ausführbaren Programm? Was passiert bei den einzelnen Schritten?

(h) Nennen Sie ein Beispiel für Operatorüberladung in der Standardbibliothek von C++.

(i) Was ist bei der Programmierung überladener Zuweisungsoperatoren in C++ zu beachten?

(j) Warum benötigt C++ im Gegensatz zu Java kein gesondertes Konzept für Interfaces?