Prüfung – Informatik 1 D-ITET, 29.01.2014
Lösungsskizze

Felix Friedrich

Kandidat/in:

Name, Vorname: .....

Legi-Nummer: .....

Ich bestätige mit meiner Unterschrift, dass ich diese Prüfung unter regulären Bedingungen ausführen konnte und dass ich die allgemeinen Bemerkungen gelesen und verstanden habe.

I confirm with my signature that I was able to take this exam under regular conditions and that I have read and understood the general guidelines.

Unterschrift: .....

Allgemeine Richtlinien und Informationen:

1. Dauer der Prüfung: 60 Minuten.
2. Erlaubte Unterlagen: Lehrbuch C++, Zusammenfassung beliebiger Länge, Ausdruck der der Präsentationen der Vorlesung. Keine elektronischen Geräte!
3. Bitte benützen Sie einen Kugelschreiber (blau oder schwarz) und keinen Bleistift. Bitte schreiben Sie leserlich. Nur lesbare Resultate werden bewertet.
4. Lösungen sind direkt auf das Aufgabenblatt zu schreiben. Ungültige Lösungen deutlich durchstreichen. Falls auf dem Aufgabenplatz nicht mehr genügend Platz ist, benutzen Sie ein separates Blatt, welches mit Ihrem Namen und Aufgabennummer beschriftet ist.
5. Falls Sie sich durch irgendjemanden oder irgendetwas gestört fühlen, melden Sie dies sofort der Aufsichtsperson.
6. Vorzeitige Abgaben sind nur bis 15min vor Prüfungsende möglich. Bleiben Sie nach Prüfungsende bitte sitzen, bis wir alle Prüfungen eingezogen haben.

General guidelines and information:

1. *Exam duration: 60 minutes.*
2. *You are allowed to use a textbook on C++, conclusions of lecture of arbitrary length and printouts of the lecture slides. No electronic devices!*
3. *Use a pen (black or blue), not a pencil. Please write legibly. We will only correct solutions that we can read.*
4. *All solutions must be written directly onto the exercise sheets. Invalid solutions need to be crossed out clearly. Should there be not enough space on the exercise sheets, use a separate sheet that is marked with name and assignment number.*
5. *If you feel disturbed by anyone or anything, immediately let the supervisor of the exam know this.*
6. *Handing in your exam preliminarily is only possible until 15 minutes before the exam ends. After the exam is finished, please stay at your desk until we collected all the exams.*

Aufgabe	1	2	3	4	5	6	$\sum$
Punkte							
(Maximal Punkte)	(15)	(18)	(17)	(15)	(18)	(17)	(100)

Aufgabe 1. (15 Punkte)

(a) Bestimmen Sie den finalen Typ (int, bool, float oder double) und Wert der folgenden C++Ausdrücke.

Determine the final type (int, bool, float or double) and value of the following C++expressions.

Voraussetzung	Ausdruck / expression	Typ / type	Wert / Value
–	`10 / 3 * 0.3f`	float	0.9
`int x;`	x = 3.0 / 2 * 5	int	7

`float x;`
`int y = 7;`
`int z = 4;`
	`x = ++y / --z ;`	float	2

`int k = 24;` `int i = 7;`	`k % i < k / i`	bool	false
–	`false - 2*true`	int	-2
–	`1u - 4 > 0`	bool	true

(b) Geben Sie für jedes der drei folgenden Code-Fragmente die Folge von Zahlen an, die das Fragment ausgibt

Provide the sequence of numbers that is generated by each of the following three code fragments.

(i)
```
for (int i=1; i<100; i*=2)
    std::cout << ++i << " ";
```

Ausgabe: 2 5 11 23 47 95

(ii)
```
int a=77;
for (int i=0; i<8; ++i)
    {
        std::cout << (a % 2) << " ";
        a /= 2;
    }
```

Ausgabe: 1 0 1 1 0 0 1 0

(iii)
```
unsigned int x = 1;
do {
    std::cout << x << " ";
    x = (5 * x + 4) % 7;
} while (x != 1);
```

Ausgabe: 1 2 0 4 3 5

Ende Aufgabe 1

Aufgabe 2. (18 Punkte)

Ergänzen Sie folgenden Code so, dass jede Permutation einer endlichen Zeichenkette ausgegeben wird. Sie können von Zeichenketten mit paarweisen verschiedenen Buchstaben ausgehen. Tipp: rekursive Implementation unter Austauschen des ersten Elementes eines Substrings mit jedem weiteren Element. Beispiel:

Complement the following code such that it outputs each permutation of a finite string. You can assume strings with mutually distinct characters. Tip: recursive implementation by exchanging the first element of a substring with each other element. Example:

```
Input : "abc"
Output: "abc acb bac bca cab cba"
```

```
#include <iostream>

void swap (char * first, char * second)
{
    char temp = *first;
    *first = *second;
    *second = temp;
}

void permute(char a[], int current_position, int size)
{
    if (current_position==size-1)
        std::cout << a << " ";
    else
    {
        for (int i = current_position; i<size; i++){
                swap(&a[i],&a[current_position]);
                permute(a, current_position+1, size);
                swap(&a[i],&a[current_position]);
        }
    }
}
int main(){
    char a[]="abc";
    permute(a,0,3);
    return 0;
}
```

Ende Aufgabe 2

Aufgabe 3. (17 Punkte)

Der folgende Datentyp `decimal` repräsentiere vorzeichenlose Dezimalzahlen mit konfigurierbarer Genauigkeit (precision Konstante). Der Konstruktor und die Print Funktion zeigen Ihnen, wie die Zahlen intern dargestellt werden. Ihre Aufgabe besteht darin, die beiden Funktion zu verstehen und den unten angegebenen += Operator zu implementieren:

The following data type `decimal` *represents unsigned decimal numbers with configurable accuracy (precision constant). The constructor and print function show you, how the numbers are stored internally. It is your task to understand the two functions and implement the += operator below.*

```
const int precision = 100;

class decimal
{
  std::vector<int> digits;

public:
  // POST: converts an unsigned value to a decimal
  decimal (unsigned value) : digits (precision)
  {
      for (int i = 0; i != precision; ++i)
      {
          digits[i] = value % 10;
          value /= 10;
      }
  }

  // POST: prints decimal value
  void print (std::ostream& os) const
  {
      int i = precision - 1;
      // find first non-zero digit
      while (!digits[i] && i) --i;
      // print in reversed order
      do os << digits[i]; while (i--);
  }
```

```
#include <vector>
#include <iostream>

const int precision = 100;

class decimal
{
	std::vector<int> digits;

public:
	// POST: converts an unsigned value to a decimal
	decimal (unsigned value) : digits (precision)
	{
		for (int i = 0; i != precision; ++i)
		{
			digits[i] = value % 10;
			value /= 10;
		}
	}

	// POST: prints decimal value
	void print (std::ostream& os) const
	{
		int i = precision - 1;

		// find first non-zero digit
		while (!digits[i] && i) --i;

		// print in reversed order
		do os << digits[i]; while (i--);
	}

	// POST: adds value of other decimal ignoring overflow
	decimal& operator += (const decimal& other)
	{
		int carry = 0;

		for (int i = 0; i != precision; ++i)
		{
			const int sum = digits[i] + other.digits[i] + carry;
			digits[i] = sum % 10;
			carry = sum / 10;
		}

		return *this;
	}
}; // end class decimal

std::ostream& operator << (std::ostream& os, const decimal& value)
```

```
{
        value.print (os); return os;
}

int main ()
{
        decimal x = 123;
        x += 1056;
        std::cout << x << '\n';
}
```

Ende Aufgabe 3

Aufgabe 4. (15 Punkte)

Das untenstehende Programm modelliert – sehr stark vereinfacht– ein Bankensystem. Es gibt Konten auf denen Geld eingezahlt und abgehoben werden kann. Damit die Bank stets weiss, wieviel Geld sie insgesamt verwaltet, wird der Gesamtwert jeweils zentral in einem Bankentotal mitgeführt. Von Zeit zu Zeit findet eine Revision statt, in welcher überprüft wird, ob sich ein Fehler in der Berechung ereignet hat, d.h. das System prüft, ob das Bankentotal noch dem Total der Beträge aller verwalteten Konten entspricht.
Fragen (Antwort jeweils maximal 3 Sätze):

The program below models – very much simplified – a banking system. There are accounts where money can be deposited and drawn out. In order to know the total amount of money managed by the bank, there is a bank total that is adapted for each transaction. From time to time a revision takes place where it is verified that computations did not contain errors. The system checks that the bank total equals the sum of amounts of all accounts.
Questions (Answer with max. 3 sentences each).

Wie beurteilen Sie die Verwendung von Fliesskommazahlen in diesem Programm?

How do you judge the use of floating point numbers in this program?

- Verwendung von Fliesskommazahlen grundsätzlich ok. Mögliche Wertebereichsprobleme müssen aber in Betracht gezogen werden, s.u.
- Alternative: Fixkommarechung mit (genügend grossen) Integern. (z.B. long long)

Nennen Sie zwei mögliche Probleme, die bei dem Programm in der aktuellen Form auftreten können und die im Zusammenhang mit der Verwendung von Fliesskommazahlen stehen.

State two possible problems that can occur in the program in its current form and that are related with the use of floating point numbers.

- Addition von sehr kleiner Zahl (1 rp) zu sehr grosser Zahl (1 Mia. Franken) führt zum Abschneiden von Nachkommastellen beim Float (24 Bit Mantisse erlaubt nur 16 Millionen verschiedene Werte)
- Vergleich zweier Fliesskommazahlen darf nicht mit $==$ durchgeführt werden.

Beschreiben Sie kurz, wie Sie das Programm anpassen könnten, so dass diese Probleme nicht mehr auftreten.

Describe shortly how you could adapt the program such that the problems do not occur any more.

- Double nehmen anstelle von Float. (53 bit Mantisse entspricht mehr als 1000 Billionen). Oder long long (64 bit fixkomma)
- Vergleich zweier Fliesskommazahlen mit Genauigkeit, etwa if (abs(a-b) $<$ eps).

```cpp
#include <iostream>
#include <vector>

struct Bank {
    float totalAmount;
};
struct Account {
    std::string owner;
    float amount;
};

Bank bank;
std::vector<Account> accounts;

void deposit(int account, double amount) {
    accounts[account].amount += amount;
    bank.totalAmount += amount;
}

void debit(int account, double amount) {
    if (amount <= accounts[account].amount) {
        accounts[account].amount -= amount;
        bank.totalAmount -= amount;
    }
}

void do_revision() {
    double revisionAmount = 0;
    for(std::vector<Account>::iterator it = accounts.begin(); it != accounts.end(); ++it) {
        revisionAmount += (*it).amount;
    }
    if (bank.totalAmount == revisionAmount) {
        std::cout << "Revision successful" << std::endl;
    } else {
        std::cout << "Revision unsuccessful, expected " << bank.totalAmount
                  << " CHF, but got " << revisionAmount << " CHF" << std::endl;
    }
}

// basic outline of the main function
int main() {
    // load accounts (...)
    while(true) {
        // execute deposits or debits (...)
        // after N deposits or debits execute revision function (...)
    }
}
```

Ende Aufgabe 4

Aufgabe 5. (18 Punkte)

Der Datentyp `PQ` im folgenden Programmtext soll Werte in aufsteigend sortierter Form speichern. Darüberhinaus sollen Werte, welche mehrfach eingegeben wurden, nur einmal vorhanden sein.
Implementieren Sie die Mitgliedsfunktion `PQ::put` entsprechend.
Beispiel einer Ausführung des Programmes (0 beendet die Eingabe)

The data type `PQ` in the following program text is supposed to store values in incrementally sorted order. Moreover, values that had been entered multiple times should only be provided once.
Implement the member function `PQ::put` accordingly.
Example of a program run (0 terminates the input):

```
Input : 1 8 2 7 2 3 1 9 3 3 2 0
Ouptut: 1 2 3 7 8 9
```

```
#include <iostream>

struct Node {
    int value;
    Node* next;
    Node (int value, Node* next): value(value), next(next) {}
};

class PQ {
    Node* first;

public:
    PQ(): first(0) {}
    void put(int value);
    bool get(int& value);
};

// insert element in queue, sorted and unique
// POST: linked lists starting at first is sorted in ascending order
//       value is contained in list once
void PQ::put(int value) {
    if (first == 0)
    {
        first = new Node(value, 0);
    }
    else
    {
        if (first->value == value) return;
        if (first->value > value)
        {
            first = new Node(value, first);
            return;
        }
        Node* prev = first;
        while (prev->next != 0 && prev->next->value < value)
        {
```

```
            prev = prev->next;
        }
        if (prev->next != 0)
            if (prev->next->value == value)
                return;
            else
                prev->next = new Node(value, prev->next);
        else
            prev->next = new Node(value, 0);
    }
}

bool PQ::get(int& value) {
    if (first)
    {
        value = first->value;
        first = first->next;
        return true;
    }
    return false;
}

int main() {
    PQ q;
    int value;
    while (std::cin >> value && value > 0)
    {
        q.put(value);
    }
    while (q.get(value))
    {
        std::cout << value << " ";
    }
}
```

Ende Aufgabe 5

Aufgabe 6. (17 Punkte)

(a) Nennen Sie die drei wichtigsten Konzepte der Objektorientierung und erklären Sie jedes Konzept sehr kurz.

Enumerate the three most important concepts of object orientation and explain each concept very shortly.

- Kapselung, Verbergen des Zustands und der Implementierungsdetails von Objekten Definition einer Schnittstelle zum Zugriff auf interne Datenstruktur Ermöglicht das Sicherstellen von Invarianten
- Verebung, Objekte können Eigenschaften von Objekten erben Abgeleitete Objekte können neue Eigenschaften besitzen oder vorhandene überschreiben Macht Code- und Datenwiederverwendung möglich

– Polymorphie, Ein Bezeichner kann abhängig von seiner Verwendung unterschiedliche Datentypen annehmen. Unterschiedliche Datentypen können bei gleichem Zugriff auf ihr gemeinsames Interface verschieden reagieren. Macht nicht invasive Erweiterung von Bibliotheken möglich

(b) Ergänzen Sie folgenden Code so, dass das Programm kompiliert werden kann und dass bei der Ausführung die Schwerpunkte der drei spezifizierten geometrischen Figuren ausgegeben werden. Die Main-Funktion darf nicht verändert werden und sollte folgende Ausgabe erzeugen:

Complement the following code such that the program can be compiled and such that a run results in output of the centroid of the specified geometric figures. The main-function may not be modified and should generate the following output:

```
centroid: 0, 2
centroid: 1.5, 2
centroid: 1, 1
```

```
#include <iostream>
#include <vector>

struct Point
{
    double x,y;
    Point(double X, double Y): x(X), y(Y) {};
    Point(): x(0), y(0) {};
};

class Figure
{
    public:
    virtual Point Centroid() const =0;
};

class Triangle:public Figure
{
    Point p1,p2,p3;
public:
    Triangle(const Point t1, const Point t2, const Point t3): p1(t1), p2(t2), p3(t3) {};

    virtual Point Centroid() const;
};

class Rectangle: public Figure
{
    Point p1, p2;
public:
    Rectangle(const Point t1, const Point t2): p1(t1), p2(t2) {};

    virtual Point Centroid() const;
};
```

```
Point Triangle::Centroid() const
{
    return Point(p1.x/3 + p2.x/3 + p3.x/3, p1.y/3 + p2.y/3 + p3.y/3);
}

Point Rectangle::Centroid() const
{
    return Point (p1.x/2 + p2.x/2, p1.y/2 + p2.y/2);
}

int main() {
    std::vector<Figure*> fig;
    typedef std::vector<Figure*>::iterator Iterator;

    fig.push_back(new Triangle(Point(-1,1), Point(1,1), Point(0,4)));
    fig.push_back(new Rectangle(Point(1,1), Point(2,3)));
    fig.push_back(new Rectangle(Point(0,0), Point(2,2)));
    for (Iterator it=fig.begin(); it != fig.end(); ++it)
    {
        Point c((*it)->Centroid());
        std::cout << "centroid: " << c.x << ", " << c.y << "\n";
    }
    // Should produce the following output:
    // centroid: 0, 2
    // centroid: 1.5, 2
    // centroid: 1, 1

}
```

Ende Aufgabe 6