Informatik I D-ITET Prüfung 4.2.2015 CAB G61Lösung *F. Friedrich*

Name, Vorname: .....

Legi-Nummer: .....

Ich bestätige mit meiner Unterschrift, dass ich diese Prüfung unter regulären Bedingungen ablegen konnte, und dass ich die allgemeinen Richtlinien 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:

General guidelines:

1. Dauer der Prüfung: 60 Minuten.

   Exam duration: 60 minutes.

2. Erlaubte Unterlagen: keine

   Permitted examination aids: none

3. Benützen Sie einen Kugelschreiber (blau oder schwarz) und keinen Bleistift. Bitte schreiben Sie leserlich. Nur lesbare Resultate werden bewertet.

   Use a pen (black or blue), not a pencil. Please write legibly. We will only correct solutions that we can read.

4. Lösungen sind direkt auf das Aufgabenblatt in die dafür vorgesehenen Boxen zu schreiben (und direkt darunter, falls mehr Platz benötigt wird). Ungültige Lösungen bitte deutlich durchstreichen! Korrekturen bei Multiple-Choice Aufgaben unmissverständlich anbringen!

   All solutions must be written directly onto the exercise sheets in the provided boxes (and directly below, if more space is needed). Invalid solutions need to be crossed out clearly. Corrections to answers of multiple choice questions must be provided without any ambiguity.

5. Falls Sie sich durch irgendjemanden oder irgendetwas gestört fühlen, melden Sie dies sofort der Aufsichtsperson.

   If you feel disturbed by anyone or anything, immediately let the supervisor of the exam know this.

6. Wir sammeln die Prüfung zum Schluss ein. Wichtig: stellen Sie unbedingt selbst sicher, dass Ihre Prüfung von einem Assistenten eingezogen wird. Stecken Sie keine Prüfung ein und lassen Sie Ihre Prüfung nicht einfach am Platz liegen. Dasselbe gilt, wenn Sie früher abgeben wollen: bitte melden Sie sich lautlos, und wir holen die Prüfung ab. Vorzeitige Abgaben sind nur bis 15 Minuten vor Prüfungsende möglich.

   We collect the exams at the end. Important: you must ensure that your exam has been collected by an assistant. Do not take any exam with you and do not leave your exam behind on your desk. The same applies when you want to finish early: please contact us silently and we will collect the exam. Handing in your exam preliminarily is only possible until 15 minutes before the exam ends.

7. Wenn Sie zur Toilette müssen, melden Sie dies einer Aufsichtsperson durch Handzeichen. Es darf zu einer Zeit immer nur eine Studentin oder ein Student zur Toilette.

   If you need to go to the toilet, raise your hand and wait for a supervisor. Only one student can go to the toilet at a time.

8. Wir beantworten keine inhaltlichen Fragen während der Prüfung. Kommentare zur Aufgabe schreiben Sie bitte auf das Aufgabenblatt.

   We will not answer any content-related questions during the exam. Please write comments referring to the tasks on the exam sheets.

Aufgabe	1	2	3	4	5	6	7	8	$\sum$
Punkte									
Maximum	6	6	6	6	6	6	6	6	48

1 Typen und Werte I *(6 Punkte)*

Geben Sie für jeden der drei Ausdrücke auf der rechten Seite jeweils C++-Typ und Wert an!

Wir merken an, dass z.B. `integer` oder `boolean` keine C++-Typen sind und als falsche Antworten gewertet werden.

Variablen `x`, `y`, und `z` seien deklariert und initialisiert wie folgt.

```
int x = 5;
int y = 2;
float z = 1.6;
```

For each of the 3 expressions on the right, provide the C++ type and value!

Note that, for example, `integer` or `boolean` are not C++ types and will be considered as incorrect answers.

Variables `x`, `y`, and `z` have been initialized as shown above.

(a) `x / y * z` 1 P

Typ/Type: `float`

(b) `x / y * z` 1 P

Wert/Value: `3.2`

(c) `++y % x-- / 1.0 * z` 1 P

Typ/Type: `double`

(d) `++y % x-- / 1.0 * z` 1 P

Wert/Value: `4.8`

(e) `x > y || y < z && z > 1.0e1` 1 P

Typ/Type: `bool`

(f) `x > y || y < z && z > 1.0e1` 1 P

Wert/Value: `true`

2 Typen und Werte II *(6 Punkte)*

Geben Sie für jeden der drei Ausdrücke auf der rechten Seite jeweils C++-Typ und Wert an!

Wir merken an, dass z.B. `integer` oder `boolean` keine C++-Typen sind und als falsche Antworten gewertet werden.

Die Arrays `a` und `b` seien deklariert und initialisiert wie folgt.

```
int a[] = {2, 4, 6};
int b[] = {1, 2, 3};
```

For each of the 3 expressions on the right, provide the C++ type and value!

Note that, for example, `integer` or `boolean` are not C++ types and will be considered as incorrect answers.

Arrays `a` and `b` have been declared and initialized as shown above.

(a) `&a[0] == &b[1]` — 1 P

Typ/Type: `bool`

(b) `&a[0] == &b[1]` — 1 P

Wert/Value: `false`

(c) `a[a[0] / b[1]]` — 1 P

Typ/Type: `int`

(d) `a[a[0] / b[1]]` — 1 P

Wert/Value: `4`

(e) `*(a + 2) / *b * 3` — 1 P

Typ/Type: `int`

(f) `*(a + 2) / *b * 3` — 1 P

Wert/Value: `18`

3 Fliessskommazahlen *(6 Punkte)*

Beantworten Sie die Fragen auf der rechten Seite, und nehmen Sie dafür an, dass Flieskommazahlen sich gemäss dem IEEE-Standard 754 für Fliesskommazahlen-Arithmetik verhalten!

Answer the questions on the right, assuming that floating point numbers behave according to the IEEE 754 standard for floating point arithmetics!

(a) **W**ahr oder **f**alsch? *true or false?* `true` 1 P

Der folgende Ausdruck hat den Wert `true`.
The following expression has value `true`:

```
1.1 - 1.0 < 0.11
```

(b) **W**ahr oder **f**alsch? *true or false?* `true` 1 P

Jede 32-bit Zahl vom Typ `unsigned int` kann ohne Wertänderung in den Typ `double` konvertiert werden.
Each 32-bit number of type `unsigned int` is convertible into the type `double`, without change in value.

(c) **W**ahr oder **f**alsch? *true or false?* `true` 1 P

Für vier Fliesskommavariablen `a`, `b`, `c`, und `d` mit beliebigem Wert gilt:
For any four floating point variables `a`, `b`, `c`, and `d`, we have:

```
a  + c * d + b == d * c + a + b
```

(d) **W**ahr oder **f**alsch? *true or false?* `true` 1 P

Eine normalisierte Fliesskommazahl kann kleiner sein als 1.
A normalized floating point number can be smaller than 1.

(e) **W**ahr oder **f**alsch? *true or false?* `true` 1 P

Es gibt Werte vom Typ `double`, für die folgendes gilt:
There are values `a` of type `double` for which the following holds:

```
a - 1 == a
```

(f) **W**ahr oder **f**alsch? *true or false?* `false` 1 P

Die beiden Literale `1.1` und `1.1f` haben den gleichen Wert.
The two literals `1.1` and `1.1f` have the same value.

4 Programmausgaben *(6 Punkte)*

Betrachten Sie folgendes Programm. Beantworten Sie die Fragen auf der rechten Seite.

```
#include<iostream>

int encode(int value)
{
    int code = 0;
    while (value > 0)
    {
        code = 10 * code + (value % 10 + value / 10 % 10) % 10;
        value /= 10;
    }
    return code;
}

void A(int value)
{
    std::cout << value;
    if (value > 0)
        A(value -1);
}

void B(int value)
{
    if (value > 0)
        B(value -1);
    std::cout << value;
}

int main ()
{
    ...
}
```

Consider the program above. Answer the questions on the right hand side!

(a) Was gibt das Programm mit folgender `main`-Funktion aus. 2 P
What is the output of the program with the following `main`-function.

```
int main () {
  std::cout << encode(56) << "\n";
  return 0;
}
```

```
15
```

(b) Was gibt das Programm mit folgender `main`-Funktion aus. 2 P
What is the output of the program with the following `main`-function.

```
int main () {
  A(4); B(4); std::cout << "\n";
  return 0;
}
```

```
4321001234
```

(c) Was gibt das Programm mit folgender `main`-Funktion aus. 2 P
What is the output of the program with the following `main`-function.

```
int main () {
  int s = 0;
  int array[3][3] = {{0, 1, 2}, {3, 4, 5}, {6, 7, 8}};

  for (int i = 0; i < 3 ; ++i) {
    for (int j = 2; j < 3 ; j++) {
      s += array[i][j];
    }
  }
  std::cout << s << "\n";
  return 0;
}
```

```
15
```

5 BNF I *(6 Punkte)*

Die folgende BNF definiert eine Sprache zur Beschreibung von Zugformationen für die Eisenbahn. Beantworten Sie die Fragen auf der rechten Seite!

```
train = open | compositions.
open = loco cars.
loco = "*" | "*" loco.
cars = "-" | "-" cars.
compositions = composition | composition compositions.
composition = "<" open loco ">".
```

The BNF above defines a language for the description of train formations for railways. Answer the questions on the right side!

(a) **W**ahr oder **f**alsch? *true or false?* `false` 1 P

Die folgende Formation ist gültig nach der BNF:
The following formation is valid according to the BNF:
`<*-*><**>`

(b) **W**ahr oder **f**alsch? *true or false?* `true` 1 P

Die folgende Formation ist gültig nach der BNF:
The following formation is valid according to the BNF:
`****-`

(c) **W**ahr oder **f**alsch? *true or false?* `true` 1 P

Die folgende Formation ist gültig nach der BNF:
The following formation is valid according to the BNF:
`<*-------------------------*>`

(d) **W**ahr oder **f**alsch? *true or false?* `false` 1 P

Die folgende Formation ist gültig nach der BNF:
The following formation is valid according to the BNF:
`<*-------*>*----`

(e) Geben Sie die kürzeste Formation an, die nach der BNF gültig ist.
Provide the shortest formation that is valid according to the BNF. 2 P

`*-`

6 BNF II *(6 Punkte)*

Wir nehmen an, dass folgende Funktionen das Gewicht (in Tonnen) einer gültigen Formation berechnen sollen. Eine Lok (ein '*') wiegt 107t, ein Wagen (ein '-') wiegt 15t. Beantworten Sie die Fragen auf der rechten Seite!

```
// POST: leading whitespace characters are extracted from is, and the
//    first non-whitespace character is returned (0 if there is none)
char lookahead (std::istream& is);
// train = open | compositions
int train (std::istream& is) {
  char c = lookahead();
  if (     A     )  B
  else return compositions(is);
}
// open = loco cars
int open (std::istream& is);
// loco = "*" | "*" loco
int loco (std::istream& is) {
    char c;
    is >> c; assert(     C     );
    int weight =  D ;
    if (     E     )  F
    return weight;
}
// cars = "-" | "-" cars
int cars (std::istream& is);
// compositions = composition | composition compositions
int compositions (std::istream& is);
// composition = "<" open loco ">"
int composition (std::istream& is);
```

We assume that the functions displayed above compute the weight (in tons) of a valid formation. A loco (a '*') weighs 107t, a wagon (a '-') weighs 15t. Answer the questions on the right hand side.

(a) Welcher Ausdruck muss bei A eingesetzt werden? 1 P
Fill in the expression for A.

A: `c == '*'`

(b) Welche Anweisung muss bei B eingesetzt werden? 1 P
Fill in the statement for B.

B: `return open(is);`

(c) Welcher Ausdruck muss bei C eingesetzt werden? 1 P
Fill in the expression for C.

C: `c == '*'`

(d) Welcher Ausdruck muss bei D eingesetzt werden? 1 P
Fill in the expression for D.

D: `107`

(e) Welcher Ausdruck muss bei E eingesetzt werden? 1 P
Fill in the expression for E.

E: `lookahead() == '*'`

(f) Welche Anweisung muss bei F eingesetzt werden? 1 P
Fill in the statement for F.

F: `weight += loco(is);`

7 Disassembly *(6 Punkte)*

Betrachten Sie folgende Prozedur, welche eine 32-bit Ganzzahl als Instruktion eines gedachten (und unvollständigen) Prozessors interpretiert. Beantworten Sie die Fragen auf der rechten Seite.

```
void disassemble(unsigned int code) {
    unsigned int op[3];
    for (int i = 0; i<3; ++i) {
        op[i] = code % 0x100;            // 0x100 = 256
        code /= 0x100;
    }
    unsigned int opcode = code % 0x80;  // 0x80 == 128
    bool carry = code / 0x80 == 1;

    int ops = 3;
    if (opcode == 9)
        std::cout << "add";
    else if (opcode == 5){
        std::cout << "out";
        ops = 2;
        }
    else{
        std::cout << "brk";
        ops = 0;
    }

    if (carry) std::cout << "c";

    for (int i = 0; i < ops; ++i){
        std::cout << " " << op[i];
    }
    std::cout << "\n";
}
```

Consider the function above that interprets a 32-bit integer as the instruction of an imagined (and incomplete) processor. Answer the questions on the right hand side.

(a) Mit wie vielen Bits ist jeder der maximal drei verwendeten Operanden codiert? 1 P
How many bits are used for the encoding of each of the maximally three operands?

```
8
```

(b) Mit wie vielen Bits ist der opcode codiert? 1 P
How many bits are used for the encoding of the opcode?

```
7
```

(c) Was ist die Ausgabe des Aufrufes `disassemble(0x050a0b0c)` ? 1 P
What is the output of the call `disassemble(0x050a0b0c)` ?

```
out 12 11
```

(d) Was ist die Ausgabe des Aufrufes `disassemble(0x090a0b0c)` ? 1 P
What is the output of the call `disassemble(0x090a0b0c)` ?

```
add 12 11 10
```

(e) Was ist die Ausgabe des Aufrufes `disassemble(0x490a0b0c)` ? 1 P
What is the output of the call `disassemble(0x490a0b0c)` ?

```
brk
```

(f) Was ist die Ausgabe des Aufrufes `disassemble(0x89010203)` ? 1 P
What is the output of the call `disassemble(0x89010203)` ?

```
addc 3 2 1
```

8 Datensätze vergleichen *(6 Punkte)*

Betrachten Sie folgenden Auszug einer Bibliothek für Personendatensätze. Die Operatoren im Code sollen Namen lexikographisch zuerst nach Nachname und dann nach Vorname vergleichen. Es gelten z.B.: Zorro Adenauer $<$ Anna Zumbühl, Mickey Mouse $<$ Minnie Mouse, Donald Trump $>$ Donald Duck. Die Operatoren ==, < (etc.) für lexikographischen Vergleich von `std::string` sind bereits in der Standardbibliothek implementiert und gegeben.

```
struct DataSet{
    std::string firstName;
    std::string lastName;
    // ...
};

bool operator == (const DataSet & left, const DataSet & right) {
  return left.firstName == right.firstName && left.lastName == right.lastName;
}
bool operator != (const DataSet & left, const DataSet & right) {
  return      A      ;
}
bool operator < (const DataSet & left, const DataSet & right) {
  return      B      ;
}
bool operator <= (const DataSet & left, const DataSet & right) {
  return      C      ;
}
bool operator > (const DataSet & left, const DataSet & right) {
  return      D      ;
}
bool operator >= (const DataSet & left, const DataSet & right) {
  return      E      ;
}
```

Consider the excerpt of a library for person data sets. The operators in the code shall compare names lexicographically by last name and first name. For example it holds: Zorro Adenauer $<$ Anna Zumbühl, Mickey Mouse $<$ Minnie Mouse, Donald Trump $>$ Donald Duck. The operators ==, < (etc.) for lexicgraphic comparison of `std::string` are already implemented in the standard library and considered given.

(a) Der ==-Operator ist bereits implementiert. Geben Sie den Ausdruck für den Ungleich-Operator in **A** an. Als **Vergleichs**operator darf **nur der ==-Operator für den Typ DataSet** verwendet werden! 1 P

The ==-operator is already implemented. Give an expression for the unqeuality operator in **A**. As **comparison** operator you may **only use the ==-Operator for type** `DataSet`!

A:
```
!(left == right)
```

(b) Geben Sie den Ausdruck für den Kleiner-Operator in **B** an. Dieser Ausdruck darf Operatoren für Stringvergleich aus der Standardbibliothek verwenden. 2 P

Give an expression for the smaller-than operator in **A**. This expression may contain operators for string comparison of the standard library

B:
```
left.lastName < right.lastName || left.lastName == right.lastName
&& left.firstName < right.firstName
```

(c) Geben Sie Ausdrücke für die restlichen Operatoren <=, > und >= an. Als **Vergleichs**operator darf **nur der Kleiner-als Operator < für den Typ DataSet** verwendet werden! 3 P

Provide expressions for the remaining operators <=, > and >=. As **comparison** operator you may **only use the smaller-operator < for type** `DataSet`!

C:
```
!(right < left)
```

D:
```
right < left
```

E:
```
!(left < right)
```