Informatik 252-0847-00 Prüfung 6. 8. 2018 Lösung *B. Gärtner*

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: Wörterbuch (für gesprochene Sprachen). Keine eigenen Notizblätter! Bei Bedarf stellen wir Ihnen weitere Blätter zur Verfügung.

   Permitted examination aids: dictionary (for spoken languages). No sheets of your own! We will give you extra sheets on demand.

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 consider 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 sind deutlich durchzustreichen! Korrekturen bei Multiple-Choice Aufgaben bitte unmissverständlich anbringen! Lösungen auf Notizblättern werden nur in Notfällen berücksichtigt. Bitte kontaktieren Sie in diesem Fall eine Aufsichtsperson.

   Solutions must be written directly onto the exam sheets in the provided boxes (and directly below, if more space is needed). Invalid solutions need to be crossed out clearly. Provide corrections to answers of multiple choice questions without any ambiguity! Solutions on extra sheets will be considered only in emergencies. In this case, please contact a supervisor.

5. Es gibt keine Negativpunkte für falsche Antworten.

   There are no negative points for false answers.

6. Störungen durch irgendjemanden oder irgendetwas melden Sie bitte sofort der Aufsichtsperson.

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

7. 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 ahead of time is only possible until 15 minutes before the exam ends.

8. Wenn Sie zur Toilette müssen, melden Sie dies einer Aufsichtsperson durch Handzeichen.

   If you need to go to the toilet, raise your hand and wait for a supervisor.

9. 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	$\sum$	Bonus
Punkte										
Maximum	9	6	6	6	8	10	9	6	60	0.25

Viel Erfolg! *Good luck!*

Aufgaben / *Tasks*

1 Typen und Werte / Types and values *(9 Punkte)*

Geben Sie für jeden der sechs Ausdrücke unten jeweils C++-Typ (0.5 P) und Wert (1 P) an! Dabei ist a ein Feld, das als `int a[] = {5, 3, 1, 2, 4};` definiert ist. *For each of the six expressions below, provide the C++ type (0.5 P) and value (1 P)! Here, a is an array defined as* `int a[] = {5, 3, 1, 2, 4};`

(a) `1+0.5` — 1.5 P

Typ/*Type*	Wert/*Value*
`double`	`1.5`

(b) `3-2*2` — 1.5 P

Typ/*Type*	Wert/*Value*
`int`	`-1`

(c) `1e4 == 16` — 1.5 P

Typ/*Type*	Wert/*Value*
`bool`	`false`

(d) `a[2]++` — 1.5 P

Typ/*Type*	Wert/*Value*
`int`	`1`

(e) `*(a+4)` — 1.5 P

Typ/*Type*	Wert/*Value*
`int`	`4`

(f) `5u / 2u` — 1.5 P

Typ/*Type*	Wert/*Value*
`unsigned int`	`2`

2 Schleifenausgaben / Loop outputs *(6 Punkte)*

Geben Sie für jedes der folgenden drei Code-Stücke die Ausgabe an, die generiert wird! Nehmen Sie für Fliesskommazahlen den Standard IEEE 754 an! Falls es sich um eine Endlosschleife handelt, schreiben Sie "Endlosschleife"!

For each of the following three code snippets, provide the output that is being generated! For floating point numbers, assume the IEEE 754 standard! In case of an infinite loop, write "infinite loop"!

(a)

```
int i = 0;
int j = 0;
while (i < 3) {
  std::cout << i << ' ';
  ++i;
  std::swap (i,j);
}
```

2 P

Ausgabe/*Output*:

```
0 0 1 1 2 2
```

(b)

```
unsigned int i = 1u;
do {
  std::cout << i << ' ';
  i = 3 * i % 5;
} while (i != 1u);
```

Ausgabe/*Output*:

```
1 3 4 2
```

2 P

(c)

```
for (bool x = false; x != true; x = !x)
  for (bool y = false; y != true; y = !y)
    std::cout << (x || y) << ' ';
```

Ausgabe/*Output*:

```
0
```

2 P

3 Primfaktorzerlegung /Prime factorization *(6 Punkte)*

Betrachten Sie das folgende Programmfragment, das eine Funktion zur Ausgabe der Primzahlzerlegung einer natürlichen Zahl n definiert. Die Primzahlzerlegung ist die aufsteigend sortierte Liste der Primfaktoren von n (mit Multiplizitäten). Nach Definition ist die Primfaktorzerlegung von $n = 0$ und $n = 1$ die leere Liste. Die Tabelle unten gibt einige Beispiele. Ihre Aufgabe ist es, das Programmfragment so zu ergänzen, dass sich eine korrekte Implementierung ergibt. In jede Lücke darf nicht mehr als ein Ausdruck oder eine Anweisung hineingeschrieben werden.

n	Primfaktorzerlegung /prime factorization
24	2 2 2 3
100	2 2 5 5
11	11
210	2 3 5 7
49	7 7

Consider the following program fragment that defines a function for computing the prime factorization of a natural number n. The prime factorization is the list of the prime factors of n (with multiplicities) in ascending order. By definition, the prime factorization of $n = 0$ and $n = 1$ is the empty list. The table above provides some examples. Your task is to complete the program fragment such that you get a correct implementation. Each gap can hold at most one expression or statement.

```
// POST: writes the prime factorization of n to std::cout
void prime_factors (unsigned int n) {
   for (unsigned int d = 2; d <= n; ++d) {
      if (n % d == 0) {
         std::cout « d « ' ';
         prime_factors (n / d);
         break;
      }
   }
}
```

4 EBNF 1 *(6 Punkte)*

Die folgende EBNF beschreibt Polynome in einer Variablen `x`. Geben Sie für jede Aussage auf dieser und der nächsten Seite an, ob sie zutrifft oder nicht.

Anmerkung: Leerschläge sind im Rahmen der EBNF bedeutungslos.

```
polynomial = monomial {op monomial}
op = "+" | "-"
monomial = number ["x^" number]
number = digit {"0" | digit}
digit = "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
```

The EBNF above describes polynomials in one variable `x`. For each statement on this and the next page, decide whether it is true or not.

Remark: *Whitespaces are irrelevant in the context of this EBNF.*

(a) **W**ahr oder **f**alsch? ***t**rue or **f**alse?* `true` 1 P

Die folgende Zeichenkette ist ein gültiges Polynom (`polynomial`) nach der EBNF:
The following string is a valid polynomial (`polynomial`) according to the EBNF:

```
2x^3 + 5x^2 + 11x^1
```

(b) **W**ahr oder **f**alsch? ***t**rue or **f**alse?* `true` 1 P

Die folgende Zeichenkette ist ein gültiges Polynom (`polynomial`) nach der EBNF:
The following string is a valid polynomial (`polynomial`) according to the EBNF:

```
17x^2 + 6x^3  - 15
```

(c) **W**ahr oder **f**alsch? ***t**rue or **f**alse?* `false` 1 P

Die folgende Zeichenkette ist ein gültiges Polynom (`polynomial`) nach der EBNF:
The following string is a valid polynomial (`polynomial`) according to the EBNF:

```
5x^2 - 13x^1 - 77x^0
```

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

1 P

Die folgende Zeichenkette ist ein gültiges Polynom (`polynomial`) nach der EBNF:
The following string is a valid polynomial (`polynomial`) according to the EBNF:

```
-4x^3 + 14x^2 - 10
```

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

1 P

Die folgende Zeichenkette ist ein gültiges Polynom (`polynomial`) nach der EBNF:
The following string is a valid polynomial (`polynomial`) according to the EBNF:

```
1
```

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

1 P

Die folgende Zeichenkette ist ein gültiges Polynom (`polynomial`) nach der EBNF:
The following string is a valid polynomial (`polynomial`) according to the EBNF:

```
6x^3 - 5x^2 - 2x
```

5 EBNF 2 *(8 Punkte)*

Wir betrachten wieder die EBNF zur Beschreibung von Polynomen aus der vorherigen Aufgabe. Unten und auf der nächsten Seite finden Sie das Fragment eines Parsers für Polynome nach der gegebenen EBNF. Ergänzen Sie die Teile des Fragment auf der nächsten Seite so, dass sich eine korrekte Implemetierung ergibt!

```
polynomial = monomial {op monomial}
op = "+" | "-"
monomial = number ["x^" number]
number = digit {"0" | digit}
digit = "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
```

We again consider the EBNF for describing polynomials from the previous task. Below and on the next page, you find the fragment of a parser for polynomials according to the EBNF. Complete the parts of the fragment on the next page such that you get a correct implementation!

```
bool polynomial (std::istream& is);
bool op (std::istream& is);
bool monomial (std::istream& is);
bool number (std::istream& is);
bool digit (std::istream& is);

// POST: if c is the next character in the input stream is, remove c
//       from is and return true, otherwise do nothing and return false
bool consume (std::istream& is, char c);

bool polynomial (std::istream& is) // polynomial = monomial {op monomial}
{
  if (!monomial (is)) return false;
  while (op (is))
    if (!monomial (is)) return false;
  return true;
}

bool op (std::istream& is) // op = "+" | "-"
{
  return consume (is, '+') || consume (is, '-');
}
```

```
bool monomial (std::istream& is) // monomial = number ["x^" number]
{
  if (!number (is))
    return false;
  if (consume (is,'x')) {
    if (!consume (is, '^'))
      return false;
    return number (is);
  }
  return true;
}

bool number (std::istream& is) // number = digit {"0" | digit}
{
  if (!digit (is))
    return false;
  while (consume (is, '0') || digit (is));
  return true;
}


bool digit (std::istream& is) // digit = "1"|"2"|"3"|"4"|"5"|"6"|"7"|"8"|"9"
{
  for (char d = '1'; d <= '9'; ++d)
    if (consume (is, d))
      return true;
  return false;
}
```

6 Berge / Mountains *(10 Punkte)*

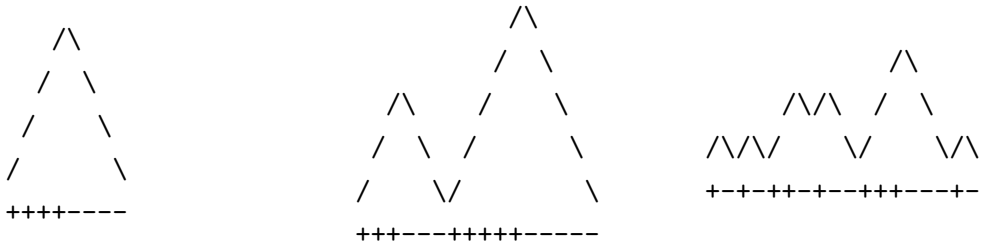
Ein Berg ist eine Folge $(x_1, x_2, \ldots, x_n), n \geq 0$ von Plus- und Minuszeichen mit den folgenden zwei Eigenschaften:

1. In jedem Präfix $(x_1, \ldots, x_k), 0 \leq k \leq n$ ist die Anzahl der Pluszeichen mindestens so gross wie die Anzahl der Minuszeichen;
2. Über die gesamte Folge ist die Anzahl der Pluszeichen gleich gross wie die Anzahl der Minuszeichen.

Interpretiert man das Pluszeichen als "Anstieg" und das Minuszeichen als "Abstieg", ergibt sich eine natürliche grafische Repräsentation eines Berges wie in den folgenden drei Beispielen.

Die Höhe eines Berges ist der maximale Überschuss von Pluszeichen in einem Präfix. Die Beispiele zeigen Berge der Höhen 4, 5 und 3.

Ihre Aufgabe ist es, die drei Funktionsdefinitionen auf der folgenden Seite gemäss der gegebenen Nachbedingungen so zu ergänzen, dass sich korrekte Implementierungen ergeben. Die Funktion `climb` darf und soll in den anderen beiden Funktionen aufgerufen werden.



A mountain is a sequence $(x_1, x_2, \ldots, x_n), n \geq 0$ of plus and minus signs with the following two properties:

1. *In every prefix $(x_1, x_2, \ldots, x_k), 0 \leq k \leq n$, the number of plus signs is at least as large as the number of minus signs;*
2. *Over the whole sequence, the number of plus signs equals the number of minus signs.*

Interpreting the plus sign as "ascent" and the minus sign as "descent", we obtain a natural graphical representation of a mountain as in the three examples above.

The height of a mountain is the maximum surplus of plus signs in a prefix. The examples depict mountains of heights 4, 5, and 3.

Your task is to complete the three function definitions on the next page according to the given postconditions such that you get correct implementations! The function `climb` can and shall be called in the other two functions.

```
// PRE: c == '+' || c == '-'
// POST: if c == '+', then h is increased by 1, otherwise h is decreased by 1;
void climb (char c, int& h) {
  assert (c == '+' || c == '-');
  1.  if (c == '+')
  2.    ++h;
  3.  else
  4.    --h;
}
// PRE: [begin, end) is a valid range with elements in {'+', '-'}
// POST: returns true if and only if the range represents a mountain
bool is_mountain (const char* begin, const char* end) {
  int h = 0; // current height
  for (const char* p = begin; p != end; ++p) {
    1.  climb (*p, h);
    2.  if (h < 0)
    3.    return false;
  }
  return h == 0;
}
// PRE: [begin, end) is a valid range representing a mountain
// POST: returns the height of the mountain represented by [begin, end)
int height (const char* begin, const char* end) {
  assert (is_mountain (begin, end));
  int h = 0; // current height
  int max_h = 0; // maximum height
  for (const char* p = begin; p != end; ++p) {
    1.  climb (*p, h);
    2.  if (h > max_h)
    3.    max_h = h;
  }
  return max_h;
}
```

7 Soziales Netzwerk / Social Network *(9 Punkte)*

Die Klasse `user` unten (nur aufgabenrelevante Members sind angegeben) repräsentiert Nutzer eines sozialen Netzwerks. Jeder Nutzer speichert eine Menge seiner Followers, kann Nachrichten an alle Followers senden und Nachrichten von anderen Nutzern empfangen. Ausserdem kann ein Nutzer anfangen oder aufhören, einem anderen Nutzer zu folgen.

Die Followers werden als Objekt vom Typ `std::set<user*>` gespeichert, also als Menge von Zeigern auf andere Nutzer. In eine Menge `s` mit zugrundeliegendem Typ T kann mit Hilfe von `s.insert (t);` der Wert von `t` eingefügt werden, wobei `t` ein Ausdruck vom Typ T ist. Mit `s.erase(t);` kann der Wert von `t` aus der Menge `s` gelöscht werden.

Ergänzen Sie das Klassenfragment auf der nächsten Seite so, dass sich eine korrekte Implementierung gemäss Vor- und Nachbedingungen ergibt!

The class `user` below (only members relevant for this task are given) represents users of a social network. Each user stores a set of followers, can send messages to all its followers and receive messages from other users. Moreover, a user can start or stop to follow another user.

Followers are stored as objects of type `std::set<user*>`, i.e. as a set of pointers to other users. Given a set `s` with underlying type T, `s.insert (t);` inserts the value of `t` into `s`, where `t` is an expression of type T. Using `s.erase(t);`, the value of `t` can be removed from the set `s`.

Complete the class fragment on the next page such that you get a correct implementation according to the pre- and postconditions!

```
#include<iostream>
#include<vector>
#include<set>
#include<string>

class user {
private:
  std::set<user*> followers;
public:
  // iterator type to go through all followers
  typedef std::set<user*>::const_iterator cit;

  // POST: *this receives message
  void receive (const std::string& message);
```

```
  // PRE: u points to a valid user
  // POST: *this starts following user *u
  void follow (user* u)
  {
    u->followers.insert (this);
  }

  // PRE: u points to a valid user
  // POST: *this stops following user *u
  void unfollow (user* u)
  {
    u->followers.erase (this);
  }

  // POST: calls member function receive with argument
  //       message for all followers of *this
  void broadcast (const std::string& message) const
  {
    1.  for (cit u = followers.begin(); u != followers.end(); ++u)
    2.    (*u)->receive (message);
  }
};
```

8 Binärdarstellung / Binary representation *(6 Punkte)*

Berechnen Sie die Binärdarstellungen $b_0.b_{-1}b_{-2}\cdots$ der unten angegebenen rationalen Zahlen $q \in (0,2)$, das heisst, schreiben Sie q jeweils in der Form

$$q = \sum_{i=0}^{\infty} b_{-i}2^{-i}, \quad b_{-i} \in \{0,1\} \text{ für alle } i.$$

Falls die Darstellung nicht endlich ist, geben Sie die exakte Periode an! Im folgenden finden Sie drei Beispiele.

q	$b_0.b_{-1}b_{-2}\cdots$
$3/2$	1.1
$7/8$	0.111
$4/3$	$1.\overline{01}$

Compute the binary representations $b_0.b_{-1}b_{-2}\cdots$ of the rational numbers $q \in (0,2)$ given below, i.e. write q in the form

$$q = \sum_{i=0}^{\infty} b_{-i}2^{-i}, \quad b_{-i} \in \{0,1\} \text{ for all } i.$$

If the representation is not finite, provide the exact period! Above, you find three examples.

q	$b_0.b_{-1}b_{-2}\cdots$
$5/8$	0.101
$7/4$	1.11
$11/7$	$1.\overline{100}$
$1/6$	$0.0\overline{01}$