Informatik 252-0847-00 Prüfung 10. 8. 2017 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	$\sum$
Punkte									
Maximum	9	6	3	8	8	6	8	12	60

Viel Erfolg! *Good luck!*

Aufgaben / *Tasks*

1 Typen und Werte *(9 Punkte)*

Geben Sie für jeden der sechs Ausdrücke unten jeweils C++-Typ (0.5 P) und Wert (1 P) an! Nehmen Sie für Fliesskommazahlen den Standard IEEE 754 an! *For each of the six expressions below, provide the C++ type (0.5 P) and value (1 P)! For floating point numbers, assume the IEEE 754 standard!*

(a) `(1 || 1) && !(1 && 1)` — 1.5 P

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

(b) `1.0e2f / 2.0e3f` — 1.5 P

Typ/*Type*	Wert/*Value*
`float`	`0.05`

(c) `0x2 * 0xf` — 1.5 P

Typ/*Type*	Wert/*Value*
`int`	`30`

(d) `1.0f == 1` — 1.5 P

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

(e) `10 % 4 * 5` — 1.5 P

Typ/*Type*	Wert/*Value*
`int`	`10`

(f) `15 / 2 / 5 * 2` — 1.5 P

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

2 Schleifenausgaben *(6 Punkte)*

Geben Sie für jedes der folgenden drei Code-Stücke die Ausgabe aus, 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)

```
double d = 2.0;
do {
    std::cout << (--d)++;
} while (d < 2.0);
```

Ausgabe/*Output*:

```
1
```

2 P

(b)

```
double d = 1.5;
while (d != 1) {
    std::cout << d << " ";
    d -= 0.1;
}
```

Ausgabe/*Output*:

```
infinite loop
```

2 P

(c)

```
int j;
for (j = 1; j <= 3; ++j)
    std::cout << j;
for (int k = 7-j; k >= 1; --k)
    std::cout << "*";
```

Ausgabe/*Output*:

```
123***
```

2 P

3 Summenberechnung *(3 Punkte)*

Betrachten Sie das folgende Programm, das eine Funktion zur Berechnung der Summe $\sum_{i=0}^{n} \frac{1}{2^i}$ definiert. Ihre Aufgabe ist es, die Definition der Funktion `sum` so zu ergänzen, dass sich eine korrekte Implementierung ergibt.

Consider the following program that defines a function for computing the sum $\sum_{i=0}^{n} \frac{1}{2^i}$ and illustrates its usage with an example. Your task is to complete the function `sum` in such a way that a correct implementation is obtained.

```
#include <iostream>
#include <cassert>

// PRE: n >= 0
// POST: returns the sum 1/(2^0) + ... + 1/(2^n)
double sum (unsigned int n) {
    assert(n >= 0);
    double s = 0;
    unsigned int c = 1;
    for (int i = 0; i <= n; ++i ) {
        s = s + 1.0 / c ;
        c = 2*c ;
    }
    return s;
}

int main() {
    std::cout << sum (3); // 1/1 + 1/2 + 1/4 + 1/8 = 1.875
    return 0;
}
```

---

4 Programmausgaben *(8 Punkte)*

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

```
#include <iostream>

void q (const int* start, const int* end) {
    const int* it = start;
    while (it < end) {
        std::cout << *it << ' ';
        it = it + *it;
    }
}

void w (const int* x) {
    std::cout << *x+2;
}

int* e (int* a, int size) {
    int b = 0;
    for (int i = 0; i < size; ++i)
        if (*(a+i) > *(a+b))
           b = i;

    return a+b;
}


int main () {
  // insert code from right hand side here
  return 0;
}
```

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

Was gibt das Programm auf der linken Seite jeweils aus, wenn man folgende Codestücke in die `main`-Funktion einfügt?

What is the respective output of the program on the left hand side when the following code pieces are inserted into the main function?

(a) 3 P

```
int a[5] = {1, 3, 5, 7, 9};
q(a, a+5);
```

```
1 3 9
```

(b) 2 P

```
int b = 5;
w(&b);
```

```
7
```

(c) 3 P

```
int c[5] = {2, 6, 8, 4, 0};
std::cout << *e(c,5);
```

```
8
```

5 Rekursion *(8 Punkte)*

Betrachten Sie die folgenden vier Funktionen `f`, `g`, `h` und `d`. Beantworten Sie die Fragen auf der nächsten Seite! *Consider the following four functions `f`, `g`, `h` and `d`. Answer the questions on the next page!*

```
#include<iostream>

int f (unsigned int x) {
    if (x == 0)
        return 0;
    if (x == 1)
        return 1;
    return f(x-1) + f(x-2);
}

//PRE: x is odd
int g (unsigned int x) {
    if (x == 1)
        return 1;
    return x * g(x-2);
}

int h (unsigned int x, unsigned int y) {
    if (y == 0) return 1;
    if (y == 1) return x;
    if (y % 2 == 0) return h (x*x, y/2);
    return x * h (x*x, (y-1)/2);
}

int d (unsigned int x) {
    if (x < 10)
        return 1;
    return 1 + d (x / 10);
}
```

(a) Betrachten Sie folgendes Hauptprogramm und geben Sie für jede Ausgabe den korrekten Wert an! *Consider the following main program and provide for each output the correct value!* 4 P

```
int main()
{
  std::cout << f (6) << std::endl;      // output: 8
  std::cout << g (7) << std::endl;      // output: 105
  std::cout << h (5, 3) << std::endl;   // output: 125
  std::cout << d (74100) << std::endl; // output: 5

  return 0;
}
```

(b) Geben Sie für jede der vier Funktionen eine Nachbedingung an, die das Verhalten der Funktion vollständig beschreibt! *For each of the four functions, write down a postcondition that completely describes the behavior of the function!* 4 P

```
// POST: returns the xth Fibonacci number
int f (unsigned int x);

// POST: returns the product of all odd numbers from 1 to x
int g (unsigned int x);

// POST: returns x^y
int h (unsigned int x, unsigned int y)

// POST: returns the number of decimal digits of x
int d (unsigned int x)
```

6 EBNF *(6 Punkte)*

Der Pizzaservice Trominoes hat ein neues Spracherkennungssystem, das telefonische Pizzabestellungen automatisch entgegennehmen kann. Damit das funktioniert, müssen die Bestellungen allerdings mittels folgender EBNF beschrieben werden. Beantworten Sie die Fragen unten und auf der nächsten Seite!

Anmerkung: Zusätzliche Leerschläge sind im Rahmen der EBNF bedeutungslos.

```
pizza = [number] size crust sauce ["with" topping {topping}]
size = "Small" | "Standard" | "Large"
crust = "classic" | "thin"
sauce = "tomato" | "creme" | "bbq"
topping = "ham" | "mozzarella" | "mushrooms"
number = "2" | "3" | "4"
```

The pizza delivery service Trominoes has a new speech recognition system that can automatically take pizza orders over the phone. For this to work, however, the orders need to be described using the EBNF above. Answer the questions below and on the next page!

Remark: *Additional 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 eine gültige Pizzabestellung (`pizza`) nach der EBNF:
The following string is a valid pizza order (`pizza`) according to the EBNF:
`Standard classic tomato with ham mozzarella`

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

Die folgende Zeichenkette ist eine gültige Pizzabestellung (`pizza`) nach der EBNF:
The following string is a valid pizza order (`pizza`) according to the EBNF:
`Small classic with mushrooms`

(c) **W**ahr oder **f**alsch? *t**r**ue or f**a**lse?* `true` 1 P

Die folgende Zeichenkette ist eine gültige Pizzabestellung (`pizza`) nach der EBNF:
The following string is a valid pizza order (`pizza`) according to the EBNF:

```
3 Large thin bbq with ham mushrooms ham
```

(d) **W**ahr oder **f**alsch? *t**r**ue or f**a**lse?* `false` 1 P

Die folgende Zeichenkette ist eine gültige Pizzabestellung (`pizza`) nach der EBNF:
The following string is a valid pizza order (`pizza`) according to the EBNF:

```
1 Small thin tomato
```

(e) **W**ahr oder **f**alsch? *t**r**ue or f**a**lse?* `false` 1 P

Die folgende Zeichenkette ist eine gültige Pizzabestellung (`pizza`) nach der EBNF:
The following string is a valid pizza order (`pizza`) according to the EBNF:

```
2 Large classic creme with
```

(f) Wie muss die Regel für `topping` abgeändert werden, damit der Kunde optional angeben kann, dass der gewünschte Mozzarella "buffalo mozzarella" sein soll? Die folgende Zeichenkette ist dann zum Beispiel eine gültige `pizza`: `"Large thin tomato with buffalo mozzarella"`. 1 P

How do you have to change the rule for `topping` in order to optionally specify that the mozzarella you want is "buffalo mozzarella"? For example the following is a valid `pizza` then: `"Large thin tomato with buffalo mozzarella"`.

```
topping = "ham" | ["buffalo"] "mozzarella" | "mushrooms", or
topping = "ham" | "mozzarella" | "buffalo mozzarella" | "mushrooms"
```

7 Zeiger und Bereiche *(8 Punkte)*

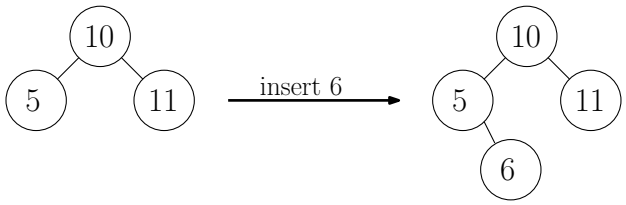
Seien $x = (x_1, \ldots, x_n), y = (y_1, \ldots, y_n) \in \{0,1\}^n$ zwei Bitvektoren. Das bitweise Und von x und y ist der Bitvektor $(x_1 \wedge y_1, \ldots x_n \wedge y_n)$, wobei $\wedge$ das logische Und ist. Der Nullvektor ist der Vektor $(0, \ldots, 0)$. Für diese Aufgabe repräsentieren wir Bitvektoren in natürlicher Weise als Felder mit zugrundeliegendem Typ `bool`. Ihre Aufgabe ist es, eine korrekte Definition der Funktion `bitwise_and` anzugeben, die einem Feld `x` das bitwise Und von `x` und einem Feld `y` zuweist und genau dann `true` zurückgibt, wenn das Ergebnis verschieden ist vom Nullvektor.

```
#include<iostream>
// PRE: [xbegin, xend) is a valid range, possibly empty. ybegin is the
//   first element of a valid range at least as long as [xbegin, xend).
// POST: the sequence x_1, x_2, ..., x_n of bits in [xbegin, xend)
//   is replaced by x_1 AND y_1, x_2 AND y_2, ..., x_n AND y_n. Returns
//   true if and only if the bitwise And is not the zero vector
bool bitwise_and (bool* xbegin, bool* xend, const bool* ybegin) {
  bool result = false;
  for (int i = 0; i < xend - xbegin; ++i) {
    *(xbegin+i) = *(xbegin+i) && *(ybegin+i); // or xbegin[i]
    if (*(xbegin+i)) result = true;
  }
  return result;
// an alternative (and actually the recommended) loop:
  bool* x = xbegin;
  const bool* y = ybegin;
  while (x != xend) {
    *x = *x && *y;
    if (*x) result = true;
    ++x;
    ++y;
  }
}
int main() {
    bool x[] = {1,0,1,0,1};
    bool y[] = {1,1,0,0,1};
    assert (bitwise_and (x,x+5,y)); // not equal to 0 0 0 0 0
    for (int i=0; i<5; ++i) std::cout << x[i] << " "; // 1 0 0 0 1
    return 0;
}
```

Let $x = (x_1, \ldots, x_n), y = (y_1, \ldots, y_n) \in \{0,1\}^n$ be two bit vectors. The bitwise And of x and y is the bit vector $(x_1 \wedge y_1, \ldots x_n \wedge y_n)$, where $\wedge$ is the logical And. The zero vector is the vector $(0, \ldots, 0)$. For this task we represent a bit vector in the natural way as an array with underlying type `bool`. Your task is to provide a correct definition of the function `bitwise_and`, which assigns to an array `x` the bitwise And of `x` and another array `y` and returns true if and only if the result is not the zero vector.

8 Binary Tree *(12 Punkte)*

Ein binärer Suchbaum ist ein Baum, in dem jeder Knoten einen Schlüssel und höchstens zwei Kinder hat, ein linkes und ein rechtes. Für jeden Knoten mit Schlüssel k has das linke Kind (wenn es existiert) einen Schlüssel $< k$ und das rechte Kind (wenn es existiert) einen Schlüssel $> k$. Ihre Aufgabe ist es, die Lücken im Code auf den nächsten beiden Seiten zu füllen. Funktion `insert`, die komplettiert werden muss, fügt einen neuen Knoten in den Baum ein. Fall der Baum leer ist, wird der neue Knoten zur Wurzel. Andernfalls wird der neue Schlüssel mit dem Schlüssel der Wurzel verglichen. Wenn er kleiner ist, wird im linken Teilbaum mit dem Einfügen weitergemacht, wenn er grösser ist, im rechten Teilbaum. Der neue Knoten wird an der Stelle eingefügt, an der der entsprechende Teilbaum leer ist. Falls ein Knoten mit dem gleichen Schlüssel bereits vorhanden ist, passiert beim Einfügen nichts. Das Beispiel unten illustriert eine Einfügeoperation. Funktion `number`, die komplettiert werden muss, gibt die Anzahl der Knoten im Baum aus. Funktion `print` gibt die Schlüssel der Knoten in einer bestimmten Reihenfolge aus. In der `main`-Funktion sollen Sie die Ausgabe eines Aufrufs von `print` angeben.



A binary search tree is a tree such that every node has a key and at most 2 children, one left and one right. For every node with key k its left child (if there is one) has a key $< k$ and its right child (if there is one) has key $> k$. Your task is to correctly complete the gaps in the code on the next two pages. Function `insert`, which has to be completed, adds a new node to the tree. If the tree is empty then the new node becomes the root. Otherwise, the new key will be compared against the key of the root: if it is smaller the insertion continues in the left subtree, and if it is larger, in the right subtree. The new node will be inserted at the position where the corresponding subtree is empty. If a node with the same key is already present, the insertion has no effect. In the example above, one insert operation is illustrated. Function `number`, which has to be completed, outputs the number of nodes in the tree. The function `print` prints the keys in a specific order. In the `main` function you have to fill in the output of this function.

```
#include <iostream>

struct node {
    node (int k)
        : key (k), left(0), right(0) {}
    int key;
    node* left;
    node* right;
};
```

```
class binary_tree {
public:
    //POST: generates an empty tree
    binary_tree ()
        : root (0) {}

    //POST: insert node at the right place
    void insert (int k) {
        if (root == 0) root = new node(k);
        node* nd = root;
        do {
            if (k == nd->key ) return;
            if (k < nd->key ) {
                if (nd->left != 0) nd = nd->left ;
                else {
                    nd->left = new node(k) ;
                    return;
                }
            }
            else {
                if (nd->right != 0) nd = nd->right ;
                else {
                    nd->right = new node(k) ;
                    return;
                }
            }
        } while (true);
    }

    int number (node* nd) {
```

```
        if (nd==0)
            return 0;
        else
            return 1 + number (nd->left) + number (nd->right);
    }

    //POST: prints the number of nodes of the tree
    int number (){
        return number (root);
    }

    void print (node* nd) {
        if (nd == 0)
            return;
        print(nd->right);
        std::cout << nd->key << " ";
        print(nd->left);
    }

    void print () {
        print (root);
    }

private:
    node* root;
};

int main() {
    binary_tree bt;
    bt.insert (10); bt.insert (5); bt.insert (11); bt.insert (6);
    bt.insert (15); bt.insert (12); bt.insert (4);

    std::cout << "# of nodes = " << bt.number() << std::endl; // output: 7
    bt.print(); // output: 15 12 11 10 6 5 4
    return 0;
}
```