



Übungsstunde W14

Informatik (RW & CBB & Statistik) – HS 23

Übersicht

Heutiges Programm

Ziele

Alte Prüfungen

Smart Pointers

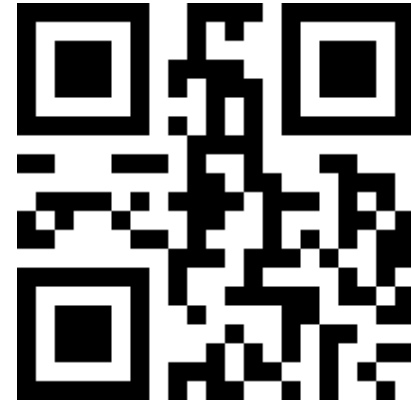
Alte Prüfung: Sommer 2021

Tipps zur Prüfungsvorbereitung

Tipps für die Lernphase

Q&A

Outro



rwko.ch/lily

1. Ziele

Ziele

- Wissen wo alte Prüfungen zu finden sind (zum Üben)
- Letzte Unklarheiten klären

Fragen/Unklarheiten?

2. Alte Prüfungen

Mit alten Prüfungen üben

- Auf der Website des Kurses sind zahlreiche alte Prüfungen zu finden
- Sie können eine gute Möglichkeit sein, für die Prüfung zu üben
- Stellt sicher, dass ihr die richtigen anschaut:
Informatik I für MPCSE



`lec.inf.ethz.ch/past_exams/`

3. Smart Pointers

Was?

Smart Pointer sind eine (speicher-)sicherere Alternative zu ("raw") Pointer

Wie?

```
#import <memory>           // to enable smart pointers
```

Anstatt raw Pointer

```
int* scptr = new int(5);
```

Kann man einfach (z. B. einen unique) Smart Pointer verwenden

```
std::unique_ptr<int> scUptr = std::make_unique<int>(5);
```

Der Rest der Verwendung ist fast identisch

- Unique Pointers erlauben kein Kopieren!

Wofür?

- Sie verwalten den Speicher automatisch!
- Führt zu weniger Speicherlecks und anderen speicherbezogenen Problemen

Fragen/Unklarheiten?

Mit ein paar Annotationen zur Erklärung
der Lösungen:)

Für die ungelösten Aufgaben siehe W14.pdf oder S21 Prüfung auf
dem Prüfungsarchiv.

(p.s. mein r sieht ein wenig komisch aus: r, oft auch e)

4. Alte Prüfung: Sommer 2021

Ich wünsche euch ganz viel Erfolg bei der Prüfungsvorbereitung!

Konversion

(1) 301: höchste Potenz ist $16^2 (=256)$ 1

=> 12D

$$\begin{array}{r} 301 \\ - 256 \\ \hline \end{array}$$

45 → höchste Potenz ist $16^1 (=32)$ 2

$$\begin{array}{r} - 32 \\ \hline \end{array}$$

13 → höchste Potenz ist $16^0 (=13)$ D

$$\begin{array}{r} - 13 \\ \hline 0 \end{array}$$

↳ Hex: nach 9 geht es mit A, B, C, D... weiter

Konvertieren Sie die Zahlendarstellungen wie angegeben. / Convert the number representations as instructed.

0110101 von binär nach dezimal / from binary to decimal

53

(1) 301 von dezimal nach hexadezimal (Basis 16) / from decimal to hexadecimal (base 16)

12D

(2) 11.625 von dezimal nach binär / from decimal to binary

1011.101

(2) zuerst vor dem Dezimalpunkt:

11 → höchste Potenz ist $2^3 (=8)$ 1

$$\begin{array}{r} - 8 \\ \hline \end{array}$$

3 → höchste Potenz ist $2^1 (=2)$ 1

$$\begin{array}{r} - 2 \\ \hline \end{array}$$

1 → höchste Potenz ist $2^0 (=1)$ 1

$$\begin{array}{r} - 1 \\ \hline 0 \end{array}$$

0 → 2^2 nicht benutzt

nach dem Dezimalpunkt:

0.625 → $2^{-1} (=0.5)$ passt 1

$$\begin{array}{r} - 0.5 \\ \hline \end{array}$$

0.125 → $2^{-2} (=0.25)$ passt nicht 0

$$\begin{array}{r} - 0.125 \\ \hline 0 \end{array}$$

$2^{-3} (=0.125)$ passt 1

=> 1011.101

Normalized Floating Point Systems

$$F^*(\beta, p, e_{\min}, e_{\max})$$

...beschreibt Zahlen der Form:

- * normalisiert ($d_0 \neq 0$)
- $\beta \geq 2$ Basis
- $p \geq 1$ Präzision (Anzahl Ziffern)
- e_{\min} kleinstmöglicher Exponent
- e_{\max} grösstmöglicher Exponent

$$\pm d_0.d_1d_2d_3 \dots d_{p-1} \cdot \beta^e$$

$$d_i \in \{0, \dots, b-1\}$$

$$d_0 \neq 0$$

$$e \in [e_{\min}, e_{\max}]$$

Zahlensysteme

Welche der folgenden Dezimalzahlen ist exakt repräsentierbar im normalisierten

Fließkommazahlensystem $F^*(\beta = 10, p = 4, e_{min} = -3, e_{max} = 3)$? / Which of the following decimal numbers is exactly representable in the normalized floating point number system

$F^*(\beta = 10, p = 4, e_{min} = -3, e_{max} = 3)$?

22.028



0.0068

alle anderen bräuchten grössere Präzision
↳ $6.8 \cdot 10^{-3}$ ✓



10.862



15432



Bewertungsmethode: **SC1/0** ?

Referenzen

```
1 int sum_swap(int& x, int y) {  
2     int temp = x;  
3     x = y;  
4     y = temp;  
5     return x + y;  
6 }  
7  
8 int a = 3;  
9 int b = 5;  
10 int c = sum_swap(a, b); → 5+3
```

x ist pass-by-reference
y ist pass-by-value
⇒ Änderungen an y
"leben" nur bis ans
Ende des Funktionsscopes
Die Variable x wird
tatsächlich verändert
und hat nachher den
Wert 5.

Welche Werte haben **a**, **b** und **c** am Ende des Codeschnippels? / What are the final values of **a**, **b** and **c**?

a =

b =

c =



Pointers

```
1 int* arr = new int[4] {2, 1, 3, 0};  
2 int* x = arr + 1;  
3 int* y = (&arr[2]) - 2;  
4  
5 std::cout << *(x + *y);
```

Welche Aussage beschreibt die Ausgabe am besten? / Which statement describes the output best?

2

Fehler (Compiler- oder Laufzeit-) / Error (compiler or runtime)

1

3

0

Bewertungsmethode: SC1/0 ?

Zeile 1: Deklariere pointer auf
int array und alloziere neues array
→ pointer **arr** zeigt auf erstes
Element des arrays

{2, 1, 3, 0}



Zeile 2: pointer **x** zeigt auf
Element hinter dem, auf das
arr zeigt (dh arr[1])

{2, 1, 3, 0}



Zeile 3: pointer **y** zeigt auf
Element an Stelle 2-2 = 0

{2, 1, 3, 0}



Zeile 5: Ausdrucken: $*(x + *y)$

{2, 1, 3, 0} $*(x + 2)$
↑↑ ↑ = 0

Pointer Constness

Es gibt zwei Arten von Constness bei Pointern:

```
const int* ptr = &a;
```

kein Schreibzugriff auf Target

a

d.h. wir dürfen den Wert des
Integers **a** *nicht* verändern

```
int* const ptr = &a;
```

kein Schreibzugriff auf Pointer

ptr

d.h. wir dürfen *nicht* ändern, wohin
der Pointer zeigt

Pointers

Welche const-Deklaration erlaubt `p++`? / Which const-declaration allows for `p++`?

`int const* p;` kein Schreibzugriff auf `target` (dh `(*p)++` geht nicht)



`int const p;` `p++` würde Wert von `p` ändern



`int* const p;` kein Schreibzugriff auf `pointer`



`int const* const p;` kein Schreibzugriff auf `pointer` & `target`



`int const* = const int*`

Frage in Stunde: `int const*` `pointer` muss auf einen `const int` zeigen

Bewertungsmethode: **SC1/0** ?

z.B. `int a = 5;`
`const int* b = &a;` ist falsch

Memory Management pt 1

```
1 int i = 42;  
2 int* p1 = &i;  
3 int* p2 = new int; *p2 = i;  
4 int* p3 = p2;
```

Welche Anweisung dealloziert den vom Code genutzten dynamischen Speicher korrekt? / Which statement correctly deallocates the dynamic memory used by the code?

delete i; kein pointer



delete p1; nicht dynamisch



delete p2; delete p3; double free



delete p3;



Rekursion

```
1 int func(int x) {  
2     if (x == 0) {  
3         return 0;  
4     }  
5     return (x % 10) + func(x / 10);  
6 }  
7  
8 std::cout << func(2806) << "\n";
```

Verbleibende Zeit 1:58:02

base case

Welche Aussage passt am besten zum Code? / Which statement matches the code best?

Terminiert nicht / Won't terminate

Gibt 0 aus / Outputs 0

Gibt 16 aus / Outputs 16

Gibt 8 aus / Outputs 8

Kompiliert nicht / Won't compile

Bewertungsmethode: SC1/0 ?

Iteratoren

- Geht auf die Prüfungssammlungs-Website
- Klickt auf den Link zum **code expert** Archiv
- Öffnet die Aufgabe der 2021.08 Math/Phys/RW Prüfung "Q12 Functions"
- Löst Task 1 und 3

Lösung zu "Q12 Functions"

```
unsigned int length_sorted_prefix(iterator begin, iterator end) {  
    if (begin == end)  
        return 0;  
    // Iterate over the while range,  
    // and count subsequent non-decreasing numbers  
    unsigned int c = 1;  
    for (; begin != end - 1 && *begin <= *(begin+1); ++begin) {  
        ++c;  
    }  
    return c;  
}
```


Lösung zu "Q12 Functions"

```
iterator find_first_occurrence(iterator first_begin, iterator first_end,
                              iterator second_begin, iterator second_end) {
    for (auto it1 = first_begin; it1 != first_end; ++it1) {
        // check that it1 could be the start of the occurrence
        auto it1_start = it1;
        auto it2 = second_begin;

        while (it1 != first_end && it2 != second_end && *it1 == *it2) {
            ++it1;
            ++it2;
        }

        if (it2 == second_end)
            return it1_start;

        it1 = it1_start ;
    }
    return first_end;
}
```

Klassen (und Structs)

- Geht zum **code expert** Archiv
- Öffnet die Aufgabe der 2021.08 Math/Phys/RW Prüfung "Q13 Clock"

Lösung zu "Q13 Clock"

```
#include "Clock24.hpp"

// PRE:  0 <= hours < 24 && 0 <= minutes < 60 && 0 <= seconds < 60
// POST:  Initializes a new clock object
Clock24::Clock24(int hours, int minutes, int seconds) {
    // TODO: Implement.
    seconds_since_midnight = 3600 * hours + 60 * minutes + seconds;
}

// POST:  returns true, if and only if the two given clocks represent
//         the point in time
bool operator==(const Clock24& first, const Clock24& second) {
    // TODO: Implement.
    // NOTE: This is not a member function.
    return !(first < second || second < first);
}
```

Lösung zu "Q13 Clock"

```
// POST: returns difference between two points in time
Duration Clock24::operator-(const Clock24& other) const {
    // TODO: Implement.
    int difference = seconds_since_midnight - other.seconds_since_midnight;

    return Duration(difference);
}

// POST: Shifts the current clock by the given duration, and returns a
//       reference to the updated receiver object.
//
Clock24& Clock24::operator+=(const Duration& duration) {
    // TODO: Implement.
    const int day = 3600 * 24;

    seconds_since_midnight =
        (seconds_since_midnight + (duration.get_seconds() % day) + day) % day;

    return *this;
}
```

Memory Management pt 2

- Geht zum **code expert** Archiv
- Öffnet die Aufgabe der 2021.08 Math/Phys/RW Prüfung "Q15 Priority Queue"

Lösung zu "Q15 Priority Queue"

```
// POST: Returns true if this node goes before the other node, according to
//       the customer queueing order described in the exercise.
bool Node::before(Node& other) const {
    return
        weight > other.weight ||
        (weight == other.weight && ticket_no < other.ticket_no);
}

// POST: Deconstructs the whole queue, which includes deallocating
//       all its nodes
PriorityQueue::~~PriorityQueue() {
    while (head != nullptr) {
        Node* tmp = head;
        head = head->next;
        delete tmp;
    }
}
```

Lösung zu "Q15 Priority Queue"

```
// PRE: Queue is not empty
// POST: Returns the number of the ticket that is first in the queue,
//        removes the corresponding node, and frees the node's memory
unsigned int PriorityQueue::pop() {
    Node* match = head;
    unsigned int ticket_no = match->ticket_no;

    head = head->next;
    delete match;

    return ticket_no;
}
```

Lösung zu "Q15 Priority Queue"

```
void PriorityQueue::insert(unsigned int _weight, unsigned int _ticket_no) {
    Node* node = new Node(_weight, _ticket_no);

    if (empty()) {
        // Case 1: queue is empty.
        head = node;
    } else if (node->before(*head)) {
        // Case 2: non-empty queue, new node becomes new head.
        node->next = head;
        head = node;
    } else {
        // Case 3: non-empty queue, new node goes further back into the queue.
        // Traverse queue, such that pred eventually points to the new node's new predecessor node.
        Node* pred = head;
        while (pred->next != nullptr && pred->next->before(*node)) {
            pred = pred->next;
        }
        node->next = pred->next;
        pred->next = node;
    }
}
```


Fragen/Unklarheiten?

5. Tipps zur Prüfungsvorbereitung

Tipps zur Prüfungsvorbereitung

- Cheat Sheet: selber schreiben oder sich inspirieren lassen (AMIV, VMP)
- üben, üben, üben
- Herausfinden, in welcher Reihenfolge ihr die Prüfung lösen wollt
- Prüfungen in 2 Stunden lösen

6. Tipps für die Lernphase

Tipps für die Lernphase

- Macht euch einen Lernplan, wenn euch das hilft
- Plant auch Pausen und Ausgleich ein
- Keep yourself sane, es geht nachher weiter

7. Q&A

Q&A

Unklarheiten?

8. Outro

Allgemeine Fragen?

Unsere letzte Übungsstunde

- Es war toll mit euch!
- Falls ihr Fragen zu Informatik, RW oder sonstigem habt: Kontaktiert mich sehr gerne unter lily.watanabe@inf.ethz.ch oder lwatanabe@ethz.ch :)

Man sieht sich an der Prüfung!

Schöne Festtage!