



Übungsstunde W02

Computer Science (CSE) – AS 23

Heutiges Programm

Kennenlernen

Organisatorisches

Integer Division & Modulo

Binärrepräsentation

Expressions und Evaluationen

Tipps zu [code]expert

Outro



`n.ethz.ch/~agavranovic`

Herzlich Willkommen!

Ziele für heute

- Uns kennenlernen
- Organisatorisches besprechen
- [code]expert einrichten
- Unklarheiten klären
- Int-Division, Operatoren und Modulo verstehen
- Zahlen in andere Basen umrechnen können
- Valide C++-Expressions erkennen
- C++-Expressions evaluieren können

1. Kennenlernen

Adel

- Hatte vor der ETH praktisch keine Erfahrungen im Programmieren

Adel

- Hatte vor der ETH praktisch keine Erfahrungen im Programmieren
- Hatte gute Note in Informatik am BPB-I

Adel

- Hatte vor der ETH praktisch keine Erfahrungen im Programmieren
- Hatte gute Note in Informatik am BPB-I
- Wird dafür sorgen, dass Ihr den BPB-I nicht wegen Informatik verkackt

Adel

- Hatte vor der ETH praktisch keine Erfahrungen im Programmieren
- Hatte gute Note in Informatik am BPB-I
- Wird dafür sorgen, dass Ihr den BPB-I nicht wegen Informatik verkackt
- Hier, um euch zu helfen

Euer Assistent

Adel

- Hatte vor der ETH praktisch keine Erfahrungen im Programmieren
- Hatte gute Note in Informatik am BPB-I
- Wird dafür sorgen, dass Ihr den BPB-I nicht wegen Informatik verkackt
- Hier, um euch zu helfen
- Trinkt gern guten Kaffee
- Fährt gern Fahrrad
- Aus Züri



Eure Kommilitoninnen und Kommilitonen

Stell dich vor!

Eure Kommilitoninnen und Kommilitonen

Stell dich vor!

- Wie heisst du?
- Woher kommst du?
- Wieso RW an der ETH?
- Hast du Erfahrung im Programmieren (insb. C++)?
- Hast du coole Hobbies oder aussergewöhnliche Interessen?

Fragen/Unklarheiten?

2. Organisatorisches



Informatik

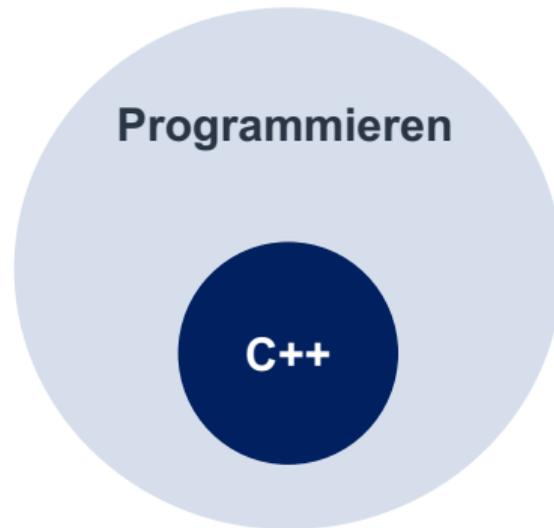
Informationen zur Organisation

Kennenlernen

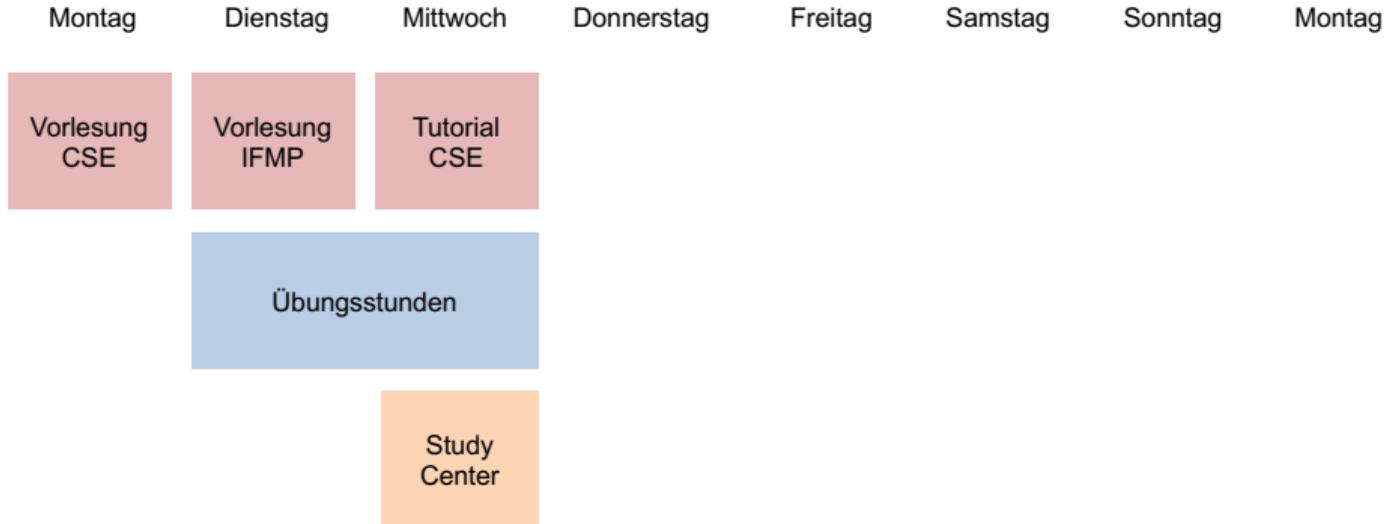
- Name?
- Welcher Teilbereich deines Studiums interessiert dich am meisten?

Ziel des Kurses

- Vorlesung
- Übungsstunde
- Wöchentliche Übungen
- Bonusaufgaben
- Study Center



Wöchentlicher Plan



Hausaufgaben auf CodeExpert lösen
(Ausgabe am Montag 6:00, Abgabe am folgenden Montag 18:00)

Wöchentliche Übungen und Bonusaufgaben

- Alle Übungen findet ihr auf [code]expert (<https://expert.ethz.ch>).
- Ihr müsst euch zuerst mithilfe dem euch gesandten Link in die Übungsgruppe einschreiben.

- **Wöchentliche Übungen:**
 - Zweck: Neuen Stoff anwenden.
 - Veröffentlichung: Montag um 6:00.
 - Deadline: Eine Woche später (Montag 18:00).
 - Erlaubt euch Erfahrungspunkte (XP) zu sammeln.

- **Bonusaufgaben** (benötigt ca. 2/3 der Erfahrungspunkte zur Freischaltung):
 - Zweck: Wissen zu verschiedenen Themen kombinieren.
 - Erlaubt euch max. +0.25 Bonus für die Endnote zu erreichen (mit 2/3 der Bonuspunkte).

Wöchentliche Übungen und Bonusaufgaben

- Alle Übungen findet ihr auf [code]expert (<https://expert.ethz.ch>).
- Ihr müsst euch für eine Übungsaufgabe in einer Übungsaufgaben-Gruppe einschreiben.
- **Wichtig!**
Benutzt zum Lösen der Übungsaufgaben **nur** das, was bereits in der Vorlesung vorgestellt wurde und was nicht von der Aufgabenstellung verboten wird. Beachtet, dass der Autograder weitere Einschränkungen (wie die Vermeidung von globalen Variablen) auferlegen kann, die nicht ausdrücklich in der Aufgabenstellung angegeben sind. Beachtet außerdem, dass Warnungen als Fehler behandelt werden. Überprüft also die Ausgabe des Autograders sorgfältig, um zu vermeiden, dass ihr 0 Punkte bekommt.
Jede Woche wird eine Zusammenfassung der in dieser Woche vorgestellten Konzepte herausgegeben.
- **Wöchentliche Aufgaben**
 - Zweck: Neugierde befriedigen.
 - Veröffentlichung: 1. Semester: 1. bis 10. November; 2. Semester: 1. bis 10. Dezember.
 - Deadline: Ende der jeweiligen Semesterferien.
 - Erlaubt euch max. +0.25 Bonus für die Endnote zu erreichen (mit 2/3 der Bonuspunkte).
- **Bonusaufgaben** (Zusammenfassung der Konzepte der Vorlesung):
 - Zweck: Wissen zu verschiedenen Themen kombinieren.
 - Erlaubt euch max. +0.25 Bonus für die Endnote zu erreichen (mit 2/3 der Bonuspunkte).

Übungsstunden

- Zweck: Vorbereitung für das Lösen von zukünftigen und vergangenen Übungen.
- Ansatz: Vor allem interaktiver Unterricht und konstruktive Diskussionen.
- Wir erwarten, dass ihr:
 - Aktiv am Unterricht teilnehmt
 - Fragen stellt, wenn ihr den Inhalt nicht versteht, oder warum wir ein bestimmtes Thema behandeln
- Es ist völlig normal, wenn ihr Fehler macht, wir sind im Lernprozess. Bitte vermeidet es, Sachen zu tun, welche andere stören könnten. Wenn eine Aufgabe zu einfach ist, helft anderen.

Study Center

- Zweck: Eine Möglichkeit, um nach individueller Hilfe zum Kurs zu fragen

IFMP:

- Zeit: Mittwoch 16:15-18:15, beginnend ab dem 27. September
- Ort: Mensa Polyterrasse
- Link: <https://studycenter.ethz.ch/>

CSE (mit MAVT geteilt):

- Zeit: Mittwoch 18:15-20:00, beginnend ab dem 4. Oktober
- Ort: HG F3

Informationen & Kontakt

- Mehr Informationen sind auf dem Informationsblatt zur Organisation zu finden.
- Für Fragen betreffend der *Vorlesungsinhalte* könnt ihr während der Vorlesung fragen.
- Für Fragen betreffend der *Übungen* könnt ihr euren TA fragen.
- Für *administrative* Fragen kontaktiert bitte den Head-TA (siehe Website für die E-Mail-Adresse).

Fragen/Unklarheiten?

Zur Übungsstunde

- Alles wird auf der meiner Website hochgeladen (meistens noch am gleichen Tag)

Zur Übungsstunde

- Alles wird auf der meiner Website hochgeladen (meistens noch am gleichen Tag)
- Wird *nicht* aufgezeichnet

Zur Übungsstunde

- Alles wird auf der meiner Website hochgeladen (meistens noch am gleichen Tag)
- Wird *nicht* aufgezeichnet
- Bitte stellt Fragen bei Unklarheiten
- Bitte beteiligt euch am Unterricht
- Bitte korrigiert mich, sobald ich Fehler mache

Zur Übungsstunde

- Alles wird auf der meiner Website hochgeladen (meistens noch am gleichen Tag)
- Wird *nicht* aufgezeichnet
- Bitte stellt Fragen bei Unklarheiten
- Bitte beteiligt euch am Unterricht
- Bitte korrigiert mich, sobald ich Fehler mache
- Ihr könnt Fragen auch direkt in euren Code bei `[code]expert` schreiben, aber das sehe ich erst *nach* der Abgabe

```
| int a = 42; // Eure Kommentare und Fragen hierhin
```

- Fragen via Mail sind immer willkommen

Zum Kurs

- Ist nicht der schwerste Kurs...

Zum Kurs

- Ist nicht der schwerste Kurs...
- ...aber einer der wichtigsten

Zum Kurs

- Ist nicht der schwerste Kurs...
- ...aber einer der wichtigsten
- Falls ihr den BPB-I nicht besteht, wird es wahrscheinlich nicht an Informatik liegen

Zum Kurs

- Ist nicht der schwerste Kurs...
- ...aber einer der wichtigsten
- Falls ihr den BPB-I nicht besteht, wird es wahrscheinlich nicht an Informatik liegen
- Prüfungen werden erst im Januar stattfinden, aber...

Zum Kurs

- Ist nicht der schwerste Kurs...
- ...aber einer der wichtigsten
- Falls ihr den BPB-I nicht besteht, wird es wahrscheinlich nicht an Informatik liegen
- Prüfungen werden erst im Januar stattfinden, aber...
- **...ÜBT WÄHREND DES SEMESTERS UND BLEIBT DRAN**

Zum Kurs

- Ist nicht der schwerste Kurs...
- ...aber einer der wichtigsten
- Falls ihr den BPB-I nicht besteht, wird es wahrscheinlich nicht an Informatik liegen
- Prüfungen werden erst im Januar stattfinden, aber...
- **...ÜBT WÄHREND DES SEMESTERS UND BLEIBT DRAN**
- ...und falls ihr doch mal abhängt: besorgt euch Hilfe, z.B. bei
 - mir
 - dem Study Center
 - euren Mitstudierenden
 - der Kanzlei

How to `[code]`expert

(Öffnet `expert.ethz.ch` auf dem Gerät eurer Wahl)

- `[code]`expert kann ein wenig pingelig sein

How to `[code]`expert

(Öffnet `expert.ethz.ch` auf dem Gerät eurer Wahl)

- `[code]`expert kann ein wenig pingelig sein
- deshalb folgt den Anweisungen sehr genau (vermeidet unnötigen Text)

How to [code]expert

(Öffnet `expert.ethz.ch` auf dem Gerät eurer Wahl)

- [code]expert kann ein wenig pingelig sein
- deshalb folgt den Anweisungen sehr genau (vermeidet unnötigen Text)
- der Autograder wird den Grossteil der Korrekturen übernehmen
- von mir bekommt ihr die letzten paar Punkte für Stil, Dokumentation, Herangehensweise, etc.

How to `[code]`expert

(Öffnet `expert.ethz.ch` auf dem Gerät eurer Wahl)

- `[code]`expert kann ein wenig pingelig sein
- deshalb folgt den Anweisungen sehr genau (vermeidet unnötigen Text)
- der Autograder wird den Grossteil der Korrekturen übernehmen
- von mir bekommt ihr die letzten paar Punkte für Stil, Dokumentation, Herangehensweise, etc.
- Textaufgaben werden vollständig manuell korrigiert

How to [code]expert

(Öffnet `expert.ethz.ch` auf dem Gerät eurer Wahl)

- [code]expert kann ein wenig pingelig sein
- deshalb folgt den Anweisungen sehr genau (vermeidet unnötigen Text)
- der Autograder wird den Grossteil der Korrekturen übernehmen
- von mir bekommt ihr die letzten paar Punkte für Stil, Dokumentation, Herangehensweise, etc.
- Textaufgaben werden vollständig manuell korrigiert
- Feedback zu den Aufgaben könnt ihr innerhalb einer Woche (von mir) erwarten (falls dringend, einfach eine Mail an mich)

Fragen/Unklarheiten?

3. Integer Division & Modulo

Der Unterschied zwischen = und ==

Assignment Operator (=)

gebraucht, um Variablen Werte zuzuweisen

Der Unterschied zwischen = und ==

Assignment Operator (=)

gebraucht, um Variablen Werte zuzuweisen

```
int a = 42; // assigns the value 42 to the variable a  
int b = 18; // assigns the value 18 to the variable b
```

Equality Operator (==)

gebraucht, um Gleichheit zwischen Variablen zu überprüfen

Der Unterschied zwischen = und ==

Assignment Operator (=)

gebraucht, um Variablen Werte zuzuweisen

```
int a = 42; // assigns the value 42 to the variable a  
int b = 18; // assigns the value 18 to the variable b
```

Equality Operator (==)

gebraucht, um Gleichheit zwischen Variablen zu überprüfen

```
(a == b) // this "expression" will equal 1 (true)  
         // or 0 (false) ("boolean")
```

Integer Division & Modulo

Integer Division & Modulo

Integer Division (a/b)

Der Compiler “ignoriert”

Dezimalstellen, wenn er (**unsigned**)

int durch (**unsigned**) **int** teilt.

Integer Division & Modulo

Integer Division (a/b)

Der Compiler "ignoriert"
Dezimalstellen, wenn er (**unsigned**)
int durch (**unsigned**) **int** teilt.

Modulo (a%b)

Division mit Rest, aber *nur* den Rest.

Integer Division & Modulo

Integer Division (a/b)

Der Compiler "ignoriert"
Dezimalstellen, wenn er (**unsigned**)
int durch (**unsigned**) **int** teilt.

7/3 ==

Modulo (a%b)

Division mit Rest, aber *nur* den Rest.

Integer Division & Modulo

Integer Division (a/b)

Der Compiler "ignoriert" Dezimalstellen, wenn er (**unsigned**) **int** durch (**unsigned**) **int** teilt.

$7/3 == 2$

$15/4 ==$

Modulo (a%b)

Division mit Rest, aber *nur* den Rest.

Integer Division & Modulo

Integer Division (a/b)

Der Compiler "ignoriert"
Dezimalstellen, wenn er (**unsigned**)
int durch (**unsigned**) **int** teilt.

7/3 == 2

15/4 == 3

16/4 ==

Modulo (a%b)

Division mit Rest, aber *nur* den Rest.

Integer Division & Modulo

Integer Division (a/b)

Der Compiler "ignoriert" Dezimalstellen, wenn er (**unsigned**) **int** durch (**unsigned**) **int** teilt.

$$7/3 == 2$$

$$15/4 == 3$$

$$16/4 == 4$$

Modulo (a%b)

Division mit Rest, aber *nur* den Rest.

$$7\%3 ==$$

Integer Division & Modulo

Integer Division (a/b)

Der Compiler "ignoriert" Dezimalstellen, wenn er (**unsigned**) **int** durch (**unsigned**) **int** teilt.

$$7/3 == 2$$

$$15/4 == 3$$

$$16/4 == 4$$

Modulo (a%b)

Division mit Rest, aber *nur* den Rest.

$$7\%3 == 1$$

$$15\%4 ==$$

Integer Division & Modulo

Integer Division (a/b)

Der Compiler "ignoriert"
Dezimalstellen, wenn er (**unsigned**)
int durch (**unsigned**) **int** teilt.

$$7/3 == 2$$

$$15/4 == 3$$

$$16/4 == 4$$

Modulo (a%b)

Division mit Rest, aber *nur* den Rest.

$$7\%3 == 1$$

$$15\%4 == 3$$

$$16\%4 ==$$

Integer Division & Modulo

Integer Division (a/b)

Der Compiler "ignoriert" Dezimalstellen, wenn er (**unsigned**) **int** durch (**unsigned**) **int** teilt.

$$7/3 == 2$$

$$15/4 == 3$$

$$16/4 == 4$$

Modulo (a%b)

Division mit Rest, aber *nur* den Rest.

$$7\%3 == 1$$

$$15\%4 == 3$$

$$16\%4 == 0$$

Integer Division & Modulo

Integer Division (a/b)

Der Compiler "ignoriert" Dezimalstellen, wenn er (**unsigned**) **int** durch (**unsigned**) **int** teilt.

$$7/3 == 2$$

$$15/4 == 3$$

$$16/4 == 4$$

Modulo (a%b)

Division mit Rest, aber *nur* den Rest.

$$7\%3 == 1$$

$$15\%4 == 3$$

$$16\%4 == 0$$

wichtige Identität

$$(a / b) * b + a \% b == a$$

Fragen/Unklarheiten?

Mal schauen, was ihr gelernt habt

- Geht auf `expert.ethz.ch`
- Logt euch ein
- Geht zu “Code Examples”
- Unter “Lecture 2: Exercise Session”, öffnet “Last Three Digits”

Mal schauen, was ihr gelernt habt

- Geht auf `expert.ethz.ch`
- Logt euch ein
- Geht zu “Code Examples”
- Unter “Lecture 2: Exercise Session”, öffnet “Last Three Digits”
- Versucht die Aufgabe zu lösen (10 Minuten)
- Wir schauen uns eure Ansätze später an

Aufgabe

Task “Last Three Digits”

Write a program which reads in an integer **a** larger than 1000 and outputs its last three digits with a space between them.

For example, if **a** = 14325, the output should be 3 2 5.

4. Binärrepräsentation

Binärrepräsentation

Binärrepräsentation

0	1	0	0	0	1	1	0
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
x128	x64	x32	x16	x8	x4	x2	x1

Binärrepräsentation

0	1	0	0	0	1	1	0
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
x128	x64	x32	x16	x8	x4	x2	x1
64		+	4			+	2
<hr/>							
70							

Bahnhofsuhr

Bahnhofsuhr

Wie spät ist es?



Bahnhofsuhr

Wie spät ist es?



Source: bahnhofsuhrsg.ch

Bahnhofsuhr

Wie spät ist es?



Source: bahnhofsuhrsg.ch

07:20:40

Binärrepräsentation

Aufgabe

Wie sieht ein Algorithmus aus, der die Binärrepräsentation einer Dezimalzahl herausfindet?

Wie würde man beispielsweise bei der Dezimalzahl 61_{10} vorgehen, um sie in Binärrepräsentation zu bringen?

Binärrepräsentation

Aufgabe

Wie sieht ein Algorithmus aus, der die Binärrepräsentation einer Dezimalzahl herausfindet?

Wie würde man beispielsweise bei der Dezimalzahl 61_{10} vorgehen, um sie in Binärrepräsentation zu bringen?

Lösung

Teile die Dezimaldarstellung durch 2 und behalte den Rest (wie eine Modulodivision). Teile dann die übrige Nummer nochmal und so weiter, bis man 0 erreicht.

Binärrepräsentation

$$61 = 2 * 30 + 1$$

$$30 = 2 * 15 + 0$$

$$15 = 2 * 7 + 1$$

$$7 = 2 * 3 + 1$$

$$3 = 2 * 1 + 1$$

$$1 = 2 * 0 + 1$$

Binärrepräsentation

$$61 = 2 * 30 + 1$$

$$30 = 2 * 15 + 0$$

$$15 = 2 * 7 + 1$$

$$7 = 2 * 3 + 1$$

$$3 = 2 * 1 + 1$$

$$1 = 2 * 0 + 1$$

Dann die letzte Spalte von unten nach oben ablesen und fertig!

$$61_{10} = 111101_2$$

Binärrepräsentation von negativen Integer

Aufgabe

Finde einen Weg, um negative Integer abzuspeichern.

Tipp:

$$n + (-n) = 0$$

Binärrepräsentation von negativen Integer

Aufgabe

Finde einen Weg, um negative Integer abzuspeichern.

Tipp:

$$n + (-n) = 0$$

Lösung

Behandle die vorderste Ziffer als das negative ihres “eigentlichen” Werts

▶ Sehr gutes Video dazu

Binärrepräsentation von negativen Integern

Aufgabe

Wie erhält man die (*signed*) `int`-Repräsentation einer beliebigen, negativen Ganzzahl $n < 0$?

Binärrepräsentation von negativen Integern

Aufgabe

Wie erhält man die (*signed*) `int`-Repräsentation einer beliebigen, negativen Ganzzahl $n < 0$?

Lösung

1. Absolutbetrag von n bestimmen
2. Absolutbetrag von n in Binärrepräsentation aufschreiben
3. Bits flippen
4. 1 addieren

Fragen/Unklarheiten?

5. Expressions und Evaluationen

Was sind Expressions?

Expressions

...sind Ausdrücke die evaluiert werden können. Sie kommen oft in Form von mathematischen Ausdrücken vor, die man dann Stück für Stück evaluiert.

Beispiel “Expression”

Evaluere die Expression¹ `5u + 5 * 3u`

¹Hier steht `u` für `unsigned int`. Der Wert muss dementsprechen gerechnet werden und es muss auf die Präzedenz der Operatoren (“Punkt vor Strich”) und die Datentypen (später mehr) geachtet werden.

Beispiel “Expression”

Evaluere die Expression¹ $5u + 5 * 3u$

$5u + 5 * 3u$

¹Hier steht `u` für `unsigned int`. Der Wert muss dementsprechen gerechnet werden und es muss auf die Präzedenz der Operatoren (“Punkt vor Strich”) und die Datentypen (später mehr) geachtet werden.

Beispiel “Expression”

Evaluere die Expression¹ $5u + 5 * 3u$

$5u + 5 * 3u$ Punkt vor Strich

¹Hier steht `u` für `unsigned int`. Der Wert muss dementsprechen gerechnet werden und es muss auf die Präzedenz der Operatoren (“Punkt vor Strich”) und die Datentypen (später mehr) geachtet werden.

Beispiel “Expression”

Evaluiere die Expression¹ $5u + 5 * 3u$

$5u + 5 * 3u$ Punkt vor Strich

$5u + (5 * 3u)$

¹Hier steht `u` für `unsigned int`. Der Wert muss dementsprechen gerechnet werden und es muss auf die Präzedenz der Operatoren (“Punkt vor Strich”) und die Datentypen (später mehr) geachtet werden.

Beispiel “Expression”

Evaluere die Expression¹ `5u + 5 * 3u`

`5u + 5 * 3u` Punkt vor Strich

`5u + (5 * 3u)` das Resultat wird zu `unsigned int`

¹Hier steht `u` für `unsigned int`. Der Wert muss dementsprechen gerechnet werden und es muss auf die Präzedenz der Operatoren (“Punkt vor Strich”) und die Datentypen (später mehr) geachtet werden.

Beispiel “Expression”

Evaluere die Expression¹ $5u + 5 * 3u$

$5u + 5 * 3u$ Punkt vor Strich

$5u + (5 * 3u)$ das Resultat wird zu **unsigned int**

$5u + 15u$

¹Hier steht **u** für **unsigned int**. Der Wert muss dementsprechen gerechnet werden und es muss auf die Präzedenz der Operatoren (“Punkt vor Strich”) und die Datentypen (später mehr) geachtet werden.

Beispiel “Expression”

Evaluere die Expression¹ `5u + 5 * 3u`

`5u + 5 * 3u` Punkt vor Strich

`5u + (5 * 3u)` das Resultat wird zu `unsigned int`

`5u + 15u` simple Addition

¹Hier steht `u` für `unsigned int`. Der Wert muss dementsprechen gerechnet werden und es muss auf die Präzedenz der Operatoren (“Punkt vor Strich”) und die Datentypen (später mehr) geachtet werden.

Beispiel “Expression”

Evaluere die Expression¹ `5u + 5 * 3u`

`5u + 5 * 3u` Punkt vor Strich

`5u + (5 * 3u)` das Resultat wird zu `unsigned int`

`5u + 15u` simple Addition

`20u`

¹Hier steht `u` für `unsigned int`. Der Wert muss dementsprechen gerechnet werden und es muss auf die Präzedenz der Operatoren (“Punkt vor Strich”) und die Datentypen (später mehr) geachtet werden.

Valide Expression erkennen

Welche der folgenden Snippets ist eine valide C++-Expression und welche nicht?

1. $1*(2*3)$

2. $(a=1)$

3. $(1$

4. $(a*3) = (b*5)$

²Wurde in [Lektion 02](#) erklärt

Valide Expression erkennen

Welche der folgenden Snippets ist eine valide C++-Expression und welche nicht?

1. $1*(2*3)$

2. $(a=1)$

3. $(1$

4. $(a*3) = (b*5)$

Lösungen

²Wurde in [Lektion 02](#) erklärt

Valide Expression erkennen

Welche der folgenden Snippets ist eine valide C++-Expression und welche nicht?

1. $1*(2*3)$

2. $(a=1)$

3. $(1$

4. $(a*3) = (b*5)$

Lösungen

- 3. ist nicht valide, da die Klammer $()$ nicht geschlossen wird

²Wurde in [Lektion 02](#) erklärt

Valide Expression erkennen

Welche der folgenden Snippets ist eine valide C++-Expression und welche nicht?

1. $1*(2*3)$

2. $(a=1)$

3. $(1$

4. $(a*3) = (b*5)$

Lösungen

- 3. ist nicht valide, da die Klammer ($()$) nicht geschlossen wird
- 4. ist nicht valide, da $(a*3)$ zu einem R-value wird, der Assignment-Operator ($=$) aber ein L-value² zu seiner Linken erwartet

²Wurde in [Lektion 02](#) erklärt

Valide Expression erkennen

Welche der folgenden Snippets ist eine valide C++-Expression und welche nicht?

1. `1*(2*3)`

2. `(a=1)`

3. `(1`

4. `(a*3) = (b*5)`

Lösungen

- 3. ist nicht valide, da die Klammer (`()`) nicht geschlossen wird
- 4. ist nicht valide, da `(a*3)` zu einem R-value wird, der Assignment-Operator (`=`) aber ein L-value² zu seiner Linken erwartet
- 1. und 2. sind valide C++-Expressions

²Wurde in [Lektion 02](#) erklärt

Valide Expression erkennen

Welche der folgenden Snippets sind L-Values und welche sind R-Values?³

1. $1*(2*3)$

2. $(a=1)$

3. $(1$

4. $(a*3) = (b*5)$

³Wurde in [Lektion 02](#) erklärt

Valide Expression erkennen

Welche der folgenden Snippets sind L-Values und welche sind R-Values?³

1. $1*(2*3)$

2. $(a=1)$

3. $(1$

4. $(a*3) = (b*5)$

Lösungen

³Wurde in [Lektion 02](#) erklärt

Valide Expression erkennen

Welche der folgenden Snippets sind L-Values und welche sind R-Values?³

1. $1*(2*3)$

2. $(a=1)$

3. $(1$

4. $(a*3) = (b*5)$

Lösungen

- 3. und 4. sind weder noch, da sie keine validen Expressions sind

³Wurde in [Lektion 02](#) erklärt

Valide Expression erkennen

Welche der folgenden Snippets sind L-Values und welche sind R-Values?³

1. $1*(2*3)$

2. $(a=1)$

3. $(1$

4. $(a*3) = (b*5)$

Lösungen

- 3. und 4. sind weder noch, da sie keine validen Expressions sind
- 1. ist ein R-Value, da der Multiplikation-Operator (*) ein R-Value zurückgibt

³Wurde in [Lektion 02](#) erklärt

Valide Expression erkennen

Welche der folgenden Snippets sind L-Values und welche sind R-Values?³

1. $1*(2*3)$

2. $(a=1)$

3. $(1$

4. $(a*3) = (b*5)$

Lösungen

- 3. und 4. sind weder noch, da sie keine validen Expressions sind
- 1. ist ein R-Value, da der Multiplikation-Operator ($*$) ein R-Value zurückgibt
- 2. ist ein L-Value, da der Assignment-Operator ($=$) ein L-Value zurückgibt

³Wurde in [Lektion 02](#) erklärt

Valide Expression erkennen

Zu was werden die Expressions evaluiert?

1. $1*(2*3)$

2. $(a=1)$

3. $(1$

4. $(a*3) = (b*5)$

Valide Expression erkennen

Zu was werden die Expressions evaluiert?

1. $1*(2*3)$

2. $(a=1)$

3. $(1$

4. $(a*3) = (b*5)$

Lösungen

Valide Expression erkennen

Zu was werden die Expressions evaluiert?

1. $1*(2*3)$

2. $(a=1)$

3. $(1$

4. $(a*3) = (b*5)$

Lösungen

- 3. und 4. können nicht evaluiert werden, da sie keine validen Expressions sind

Valide Expression erkennen

Zu was werden die Expressions evaluiert?

1. $1*(2*3)$

2. $(a=1)$

3. $(1$

4. $(a*3) = (b*5)$

Lösungen

- 3. und 4. können nicht evaluiert werden, da sie keine validen Expressions sind
- 1. evaluiert zu 6, also das Ergebnis der simplen Multiplikation

Valide Expression erkennen

Zu was werden die Expressions evaluiert?

1. $1*(2*3)$

2. $(a=1)$

3. $(1$

4. $(a*3) = (b*5)$

Lösungen

- 3. und 4. können nicht evaluiert werden, da sie keine validen Expressions sind
- 1. evaluiert zu 6, also das Ergebnis der simplen Multiplikation
- 2. evaluiert zu 1, da **a** zurückgegeben wird, dem gerade davor der Wert 1 zugewiesen wurde

Fragen/Unklarheiten?

6. Tipps zu `[code]`expert

Tipps für [code] expert

- Früh genug anfangen
- Zusammen daran arbeiten (aber nicht abschreiben!)

7. Outro

Allgemeine Fragen?

Bis zum nächsten Mal

Schöne Woche!