ETH Zürich

Informatik I
Übungsstunde 04

Johannes Odenthal

ETZ J 91
cs.jodenthal.com

Hausaufgabenreview · Exercise 02

Equivalent Resistance · Ausschnitte aus main

Don’t

```cpp
int a;
int b;
std::cin >> a;
std::cin >> b;
int e;
e = a + b;
```

Do

```cpp
int R1;
int R2;
std::cin >> R1 >> R2;
const int R12 = R1 + R2;
```

Hausaufgabenreview · Exercise 03

From Natural Language to C++ · task4

Don’t

```cpp
bool task4(bool a, bool b, bool c) {
  return ((!a && !b) && !c)
      || ((a || b) && c);
}
```

Do

```cpp
bool task4(bool a, bool b, bool c) {
  return (!a && !b) != c;
}
```

Hausaufgabenfragen

Fragen zu Exercise 2 oder 3?

Loops

Loop Correctness

`n` ist eine ganze Zahl vom Typ `int`.

for

```
for (int i = 1;
     i <= n; ++i) {
  std::cout << i
       << "\n";
}
```

while

```
int i = 0;
while (i < n) {
  std::cout
       << ++i
       << "\n";
}
```

do–while

```
int i = 1;
do {
  std::cout
       << i++
       << "\n";
} while (i <= n);
```

Für welche Werte von `n` unterscheiden sich die Ausgaben?

Was passiert bei n = 0?

for

```cpp
for (int i = 1;
     i <= n; ++i) {
  std::cout << i
       << "\n";
}
```

while

```cpp
int i = 0;
while (i < n) {
  std::cout
       << ++i
       << "\n";
}
```

do–while

```cpp
int i = 1;
do {
  std::cout
       << i++
       << "\n";
} while (i <= n);
```

Eingabe	for	while	do–while
`n = 3`	1, 2, 3	1, 2, 3	1, 2, 3
`n = 0`	keine Ausgabe	keine Ausgabe	1

Was passiert bei n = INT_MAX?

`INT_MAX` ist der grösste darstellbare Wert von `int`.

for

```
for (int i = 1;
     i <= n; ++i) {
  std::cout << i
       << "\n";
}
```

Integer-Überlauf Nach der letzten Ausgabe erhöht `++i` nochmals.

while

```
int i = 0;
while (i < n) {
  std::cout
       << ++i
       << "\n";
}
```

Stoppt korrekt `++i` erreicht `INT_MAX`; danach ist `i < n` falsch.

do–while

```
int i = 1;
do {
  std::cout
       << i++
       << "\n";
} while (i <= n);
```

Integer-Überlauf `i++` erhöht bereits bei der letzten Ausgabe.

Vorzeichenbehafteter Integer-Überlauf: undefined behavior.

for ⟶ while

Die Ausgabe soll für dasselbe `n` gleich bleiben.

Ausgangscode

```
for (int i = 0; i < n;
     ++i) {
  std::cout << i << " ";
}
```

Lösung

```
{
  int i = 0;
  while (i < n) {
    std::cout << i << " ";
    ++i;
  }
}
```

Initialisierung davor · Bedingung im Kopf · Update am Rumpfende

`n = 3`: Ausgabe `0 1 2`

while → for

Die Ausgabe soll für dasselbe `n` gleich bleiben.

Ausgangscode

```cpp
int i = n;
while (i > 0) {
  std::cout << i << " ";
  --i;
}
```

Lösung

```cpp
int i = n;
for (; i > 0; --i) {
  std::cout << i << " ";
}
```

`i` bleibt ausserhalb deklariert; Bedingung und Update stehen im Kopf.

`n = 3`: Ausgabe `3 2 1`

do–while ⟶ for

Die Ausgabe soll für dasselbe `n` gleich bleiben.

Ausgangscode

```
1 int i = 0;
2 do {
3   std::cout << i << " ";
4   ++i;
5 } while (i < n);
```

Lösung

```
1 int i = 0;
2 std::cout << i << " ";
3 ++i;
4 for (; i < n;) {
5   std::cout << i << " ";
6   ++i;
7 }
```

Erster Durchlauf vor dem Test; die `for`-Schleife übernimmt den Rest.

`n = 0`: Ausgabe `0` · `n = 3`: Ausgabe `0 1 2`

Fragen?

Computing Taylor Series

Sinus als unendliche Summe

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \dots$$

Was ändert sich von Summand zu Summand?

Vorzeichen: $+, -, +, -, \dots$

Exponent: $1, 3, 5, 7, \dots$ — jeweils $+2$

Nenner: $1!, 3!, 5!, 7!, \dots$ — Fakultät des Exponenten

Fakultät

$$n! = \prod_{k=1}^{n} k$$

Berechne die Fakultät einer nichtnegativen ganzen Zahl `n`.

Beispiel: `n = 5` ⟶ `120`

Welche Variablen und Startwerte brauchen wir?

Produkt `result = 1.0` · Zähler `k = 1`

Bei `n = 0`: kein Durchlauf, Ergebnis `1`.

Fakultät mit einer Schleife

```
double result = 1.0;
for (int k = 1; k <= n; ++k) {
  result *= k;
}
```

Start: Produkt ist `1`, nicht `0`.

Zähler: von `1` bis einschliesslich `n`.

Update: das bisherige Produkt mit `k` multiplizieren.

$$5!: \quad 1 \to 1 \to 2 \to 6 \to 24 \to 120$$

Die Fakultät als Funktion

```
1 double factorial(int n) {
2   double result = 1.0;
3   for (int k = 1; k <= n; ++k) {
4     result *= k;
5   }
6   return result;
7 }
```

```
1 double value = factorial(5);  // value: 120
```

Taylor Series

CodeExpert

Fragen?

Scopes

Scopes · Sichtbarkeit

Dieser Code enthält einen Fehler.

```
1 int x = 2;
2 if (x < 7) {
3   int a = 8;
4   std::cout << a;
5 }
6 std::cout << a;
```

Warum kompiliert dieser Code nicht?

`a` ist nach dem Ende des `if`-Blocks nicht sichtbar.

Die letzte Ausgabe in den Block verschieben — oder `a` ausserhalb deklarieren.

Shadowing · Code Tracing

```
 1 int main() {
 2   int x = 2;
 3   int a = 2;
 4   if (x < 7) {
 5     int a = 8;
 6     std::cout << a;
 7   }
 8   std::cout << a;
 9   return 0;
10 }
```

a_1: aussen · a_2: innen

Schritt	Auswertung	a_1	a_2	Ausg.
Init	x = 2	—	—	
Init	äusseres a = 2	2	—	
Test	2 < 7 ⟶ true	2	—	
Init	inneres a = 8	2	8	
Ausgabe	inneres a lesen	2	8	8
Blockende	inneres a endet	2	—	8
Ausgabe	äusseres a lesen	2	—	82

Scopes in einer Schleife

```
1 int sum = 0;
2 for (int i = 0; i < 5; ++i) {
3   int a;
4   std::cin >> a;
5   sum += a;
6 }
```

Wo sind die Namen sichtbar?

sum · auch nach der Schleife

i · Kopf und Rumpf der Schleife

a · ab Deklaration im Rumpf

Bei jedem Durchlauf entsteht ein neues `a`.

Fragen?

Nachschlagen

Rückgabewert und Ausgabe

Eine Zahl zurückgeben

```
1 double area(double w,
2             double h) {
3   return w * h;
4 }
```

```
1 double result =
2     area(3.0, 4.0);
```

Eine Zahl ausgeben

```
1 void print_value(
2     double value) {
3   std::cout << value
4       << "\n";
5 }
```

```
1 print_value(12.0);
```

break und continue

break

```
for (int i = 1; i <= 5;
     ++i) {
  if (i == 3) {
    break;
  }
  std::cout << i;
}
```

continue

```
for (int i = 1; i <= 5;
     ++i) {
  if (i == 3) {
    continue;
  }
  std::cout << i;
}
```

Welche Ausgabe entsteht jeweils?

break: `12` · **continue:** `1245`

switch

```
1 switch (door) {
2   case 1:
3   case 3:
4     std::cout << "Wrong choice\n";
5     break;
6   case 2:
7     std::cout << "You won\n";
8     break;
9   default:
10    std::cout << "Unknown door\n";
11 }
```

Was wird bei `door = 2` ausgegeben?

Ausgabe: `You won`