

Version C++ 23

## Übungsblatt 4

### Aufgabe 1 - Referenzen

Gegeben seien folgende überladene Funktionen in der Programmiersprache C++:

(a) 

```
int inkrementiere(int &v)
{
    return ++v;
}
```

(b) 

```
int inkrementiere(const int &v)
{
    return v+1;
}
```

1. Welche Implementierung (a), (b) wird durch welchen Funktionsaufruf (i) und (ii) ausgeführt?

(i) 

```
int x = 10;
int y = inkrementiere(x);
```

 → (a), da x nicht const ist.

(ii) 

```
int z = inkrementiere(10);
```

 → (b), da 10 ein Literal, und somit const, ist.

2. Wie lauten die Werte in x, y und z nach der Ausführung der Aufrufe in (i) und (ii) ?

*x=11, y=11, z=11*

3. Worin unterscheidet sich die Implementierung in a) von der Implementierungen in b)? Warum ist in b) eine andere Implementierung notwendig?

### Aufgabe 2 - Werte, Zeiger und Referenzen

<pre>double a {2.3};</pre>	<i>R-Wert</i>	<i>2,3</i>
<pre>double *b {a};</pre>	<i>L-Wert</i>	<i>9</i>
<pre>double c {10.3-0.3};</pre>	<i>R-Wert</i>	<i>10</i>
<pre>float &amp;d {0};</pre>	<i>R-Wert</i>	<i>falsch</i>
<pre>double &amp;e {a};</pre>	<i>L-Wert</i>	<i>a=2,3</i>
<pre>double *f {&amp;e};</pre>	<i>L-Wert</i>	<i>2,3</i>
<pre>short *g {b+1};</pre>	<i>R-Wert</i>	<i>falsch</i>
<pre>const double &amp;h {4.0+a};</pre>	<i>R-Wert</i>	<i>6,3</i>
<pre>double *i = &amp;(c-0.5);</pre>	<i>R-Wert</i>	<i>falsch?</i>
<pre>const int &amp;j {20};</pre>	<i>R-Wert</i>	<i>20</i>
<pre>int &amp;&amp;k = j + 10;</pre>	<i>R-Wert</i>	<i>30</i>

Listing 1: Initialisierung von Variablen in C++.

- Geben Sie in Listing 1 für **jeden** Ausdruck, der die Variablen *a-k* initialisiert, an, ob es sich um einen L-Wert oder R-Wert handelt.
- Markieren Sie richtige und falsche Quelltext-Zeilen (C++ 23) in Listing 1. Geben Sie wenn möglich den Wert an, der in den Variablen *a - k* gespeichert oder - im Falle von Zeigern und Referenzen - referenziert wird.

### Aufgabe 3 - Dynamische Speicherverwaltung

1. Geben Sie mindestens drei Nachteile der dynamische Speicherverwaltung in C++ an.
2. Welche Vorteile haben intelligente Zeiger (Smart Pointer)? Nennen Sie mind. drei.
3. Gegeben sei die Klasse `Quadrat` mit Konstruktor `Quadrat(int groesse)` und einer Member-Funktion `void zeichnen()`. In unten stehendem Quelltext zum Zeichnen eines Quadrates (der Seitenlänge 5) haben sich kleine Fehler eingeschlichen. Korrigieren Sie diese!

```
Quadrat *quad {new Quadrat[5]}; Quadrat *quad (5);  
quad->zeichnen(); quad.zeichnen();  
delete Quadrat; delete quad;
```

Listing 2: Dynamische Objekterzeugung

4. Schreiben Sie den Quelltext um und verwenden Sie den Smart Pointer `std::unique_ptr` aus der C++-Standardbibliothek (C++ 23).

3]

1) Man muss sich selbst um Speicherverwaltung /-freigabe  
• Gefahr von Memory leaks  
• ist von der Laufzeit langsamer

2) Speicherbedarf ist erst zur Laufzeit bekannt  
• Objekte werden nicht direkt am Ende des Wertebereichs gelöscht  
• Deklaration und Initialisierung kann an verschiedenen Orten statt finden