1 Makro: MAX

1.1 Aufgabe

Sie haben folgendes Makro mit dem Präprozessor definiert:

```
1 #define MAX(a, b) ((a) < (b) ? (b) : (a))
```

Und in `main()` folgende Anweisung geschrieben:

```
1 int x = 5, y = 10;
2 int z = MAX(x++, y++);
```

Welches Werte erhalten Sie für `x`, `y`, und `z`? Entspricht es dem, was Sie erwartet haben? Wenn nein, korrigieren Sie das Programmfragment.

1.2 Lösung

Resultat:

```
1 int z = ((5++) < (10++) ? (11++) : (6++)); // x = 6, y = 12, z = 11
```

Erwartet wird:

```
1 int z = ((5) < (10) ? (10++) : (5++)); // x = 6, y = 11, z =  10
```

Korrektur:

```
1 int x = 5, y = 10;
2 int z = MAX(x, y);
3 x++;
4 y++;
```

Erklärung: Obwohl es beim Aufruf so aussieht, dass die Expression `y++` nur einmal evaluiert wird, geschieht dies doppelt, weil die Implementation des Makros die Expression `(b)` doppelt verwendet. Für `(a)` gilt dies in diesem Beispiel nicht, da die Evaluierung des Conditionals `(a)<(b)` in diesem Fall in `(b)` resultiert.

2 Speicherbedarf

2.1 Aufgabe

Messen Sie auf Ihrem System den Speicherbedarf der primitiven Datentypen und geben Sie die Resultate tabellarisch aus:

```
char        1 Byte
short       2 Bytes
...
```

Tipp: Die Funktion `sizeof()` könnte weiterhelfen.

Knacknuss: Verwenden Sie ein parametrisiertes Makro zur starken Vereinfachung des Programmcodes.

2.2 Lösung

```cpp
#pragma once

#include <iostream>
#include <iomanip>

#define TYPE_SIZE_MACRO(type) \
  std::setw(12) << std::left << #type << sizeof(type) << \
  (sizeof(type) == 1 ? " Byte" : " Bytes")

void TestTypeSize() {
  std::cout << "TestTypeSize" << std::endl;
  std::cout << "------------" << std::endl;

  std::cout << TYPE_SIZE_MACRO(bool) << std::endl;
  std::cout << TYPE_SIZE_MACRO(char) << std::endl;
  std::cout << TYPE_SIZE_MACRO(short) << std::endl;      // NOLINT[runtime/int]
  std::cout << TYPE_SIZE_MACRO(int) << std::endl;
  std::cout << TYPE_SIZE_MACRO(long) << std::endl;       // NOLINT[runtime/int]
  std::cout << TYPE_SIZE_MACRO(long long) << std::endl;  // NOLINT[runtime/int]
  std::cout << TYPE_SIZE_MACRO(float) << std::endl;
  std::cout << TYPE_SIZE_MACRO(double) << std::endl;
  std::cout << TYPE_SIZE_MACRO(long double) << std::endl;

  std::cout << std::endl;
}
```