# Aufgaben 1

* Deklariere einige Variablen mit den Standardtypen von C99 (<stdint.h>), resp. C++11 (<cstdint>).
* Benutze „normale“ Initialisierung der Variablen und Funktionsnotation.
* Was bietet die Klasse String? Welches sind die Members, welche Funktionen
* if (a + 1) ...  
  Eigentlich müsste der Klammerinhalt vom Typ bool sein. Was passiert bei folgendem Code?  
  bool b;  
  b = a + 1;
* Funktion change(int& x, int& y), was passiert wenn  
  int& temp = x;  
  x = y;  
  y = temp;
* Arrays: gibt es zur Laufzeit eine Bereichsüberprüfung? Was passiert, wenn die Arraygrenzen überschritten werden?

Array:   
An array is a leftover from C. C arrays are about as primitive as you can get. Just a sequence of elements with contiguous addresses. There is no bounds checking because it is simply exposing raw memory. Implementing a robust bounds-checking mechanism would have been almost impossible in C.

In C++, bounds-checking is possible on class types. But an array is still the plain old C-compatible one. it is not a class. Further, C++ is also built on another rule which makes bounds-checking non-ideal. The C++ guiding principle is "you don't pay for what you don't use". If your code is correct, you don't need bounds-checking, and you shouldn't be forced to pay for the overhead of runtime bounds-checking.