Universität Bielefeld
Technische Fakultät

AG Genominformatik
Dr. Lucas Robidou

The Rust programming language
Summer 2024 / 2025

Exercises

1 Const

1. Write the `make_array` function, so that it is usable to initialize `ZERO_ARRAY`.

```
const SIZE: usize = 5;
const ZERO_ARRAY: [u8; SIZE] = make_array();

/// Returns an array of ‘SIZE‘ elements, all set to 0.
fn make_array() -> [u8; SIZE] {

}
```

2. Consider the following function:

```
const fn square(n: usize) -> usize {
    n * n
}

pub fn main() {
    println!("{}", square(5));
}
```

   In this context, is `square` executed at runtime? At compile time?

3. Modify the function of the first exercise, so that it takes a value and a size, and returns an array of that size, all elements being set to that value.

4. Write a function named `array_two_bytes` that takes a value of any type, and returns an array of 4 elements, each initialized to said value. The twist: the function should only compile if the size of the given value is exactly 2 bytes.

5. You can use assert! at:

   - runtime
   - runtime, but only in debug mode (with `debug_assert`)
   - compile time

   There is also another one, `assert_unchecked`, which never panics.

   (**Never** use `assert_unchecked` without reading the documentation before.)

   Before looking at the documentation, why would anyone want to use an assertion that will always pass?