FH University of Applied Sciences TECHNIKUM WIEN

# String

Implement a universally usable string library. "Universal" in this context means that the library should work with different compilers and platforms and behave consistently across all platforms.

Create appropriate test cases to test your string class. The provided test cases do not cover all possible scenarios.

Pay attention to performance and memory management during implementation. This does not mean that you need to optimize your class down to the last millisecond, but unnecessary allocations, copies, or computations may result in point deductions.

Ensure **const correctness** and write **meaningful comments** where appropriate.

## Task Part 1:

- Create a string using **const char***
- Concatenate strings using an **append()** member function
- Allow access to the internally managed character array using the **c_str()** member function (c string equivalent)
- Implement a getter that returns the length of the string – **length()**

## Task Part 2:

- Implement a **copy constructor**
- Implement a **copy assignment operator**
- Implement a **move constructor**
- Implement a **move assignment operator**
- Implement a **+= operator** and a **+ operator** (**two overloads each**)
- Implement a **conversion function** for const char*

## Bonus Challenge:

- Implement a bidirectional **iterator** for the string class
  - The iterator should be implemented using a **nested class**
  - Provide **begin()** and **end()** methods
  - Overload appropriate operators