main.cpp

```cpp
#include <iostream>
#include <string>

using namespace std;

int main() {
  string name;
  cin >> name;
  cout << "Hello "
       << name << "!" << endl;
}
```

Kompilieren auf der Konsole

```
g++ -Wall main.cpp
./a.out < sample.in
./a.out < sample.in | diff - sample.out

g++ -std=gnu++17 -O0 -g -Wall -Wextra -Wconversion main.cpp \
&& ./a.out < sample.in | diff - sample.out
```

Cheatsheet

```cpp
// einmal mit allem bitte
#include <bits/stdc++.h>
// wer will schon immer std:: schreiben
using namespace std;
// schreibfaul aber wissen was drin ist
typedef int32_t i32;
typedef int64_t i64;
typedef uint32_t u32;
typedef uint64_t u64;
// besser ist das
#define float fliesskommazahlensindboese
#define double einfachnichtbenutzen
// und los
int main() { ... }
```

CMakeLists.txt

```
cmake_minimum_required(VERSION 3.22)
project(APW)

set(CMAKE_CXX_FLAGS_DEBUG "-O0 -g -Wall -Wextra -Wconversion")
set(CMAKE_CXX_FLAGS_RELEASE "-O3 -g -DNDEBUG -march=native -mtune=native")

set(CMAKE_CXX_STANDARD 17)

add_executable(w1_a1 w1/a1.cpp)
add_executable(w1_a2 w1/a2/main.cpp)
```

Java FastIO

```java
public class FastIO {
  static BufferedReader r = new BufferedReader(new InputStreamReader(System.in));
  static StringTokenizer tokens = new StringTokenizer("");
  static PrintStream out = new PrintStream(new BufferedOutputStream(System.out, f

  static boolean hasNext() throws IOException {
    while (!tokens.hasMoreTokens()) {
      String line = r.readLine();
      if (line == null) return false;
      tokens = new StringTokenizer(line);
    }
    return true;
  }

  static String next() throws IOException {
    hasNext();
    return tokens.nextToken();
  }

  static int nextInt() throws IOException {
    return Integer.parseInt(next());
  }
}
```