Time Duration

The time duration is a measure of how many ticks have passed since a certain time point. The implementation is presented in this lesson.

WE'LL COVER THE FOLLOWING

Time duration

Time duration is the difference between the two time-points. Time duration is measured in the number of ticks.

```
template <class Rep, class Period = ratio<1>> class duration;
```

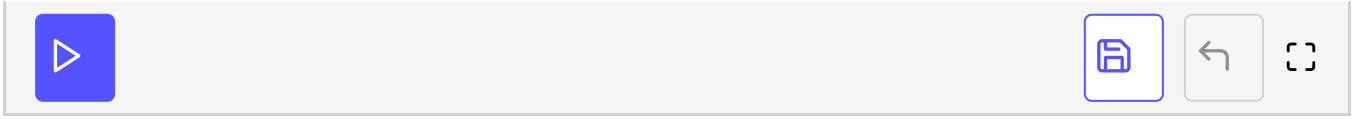
If `Rep` is a floating point number, the time duration supports fractions of ticks. The most important time durations are predefined in the chrono library:

```
typedef duration<signed int, nano> nanoseconds;
typedef duration<signed int, micro> microseconds;
typedef duration<signed int, milli> milliseconds;
typedef duration<signed int> seconds;
typedef duration<signed int, ratio< 60>> minutes;
typedef duration<signed int, ratio<3600>> hours;
```

How long can a time duration be? The C++ standard guarantees that the predefined time durations can store +/- 292 years. You can easily define your own time duration like a German school hour: `typedef std::chrono::duration<double, std::ratio<2700>> MyLessonTick`. Time durations in natural numbers have to be explicitly converted to time durations in floating pointer numbers. The value will be truncated:

```
// duration.cpp
#include <iostream>
#include <chrono>
```

```
#include <ratio>
using namespace std;
using namespace std::chrono;

template <class Rep, class Period = ratio<1>> class duration;

int main(){
  typedef std::chrono::duration<long long, std::ratio<1>> MySecondTick;
  MySecondTick aSecond(1);

  milliseconds milli(aSecond);
  std::cout << milli.count() << " ms\n";        // 1000 milli

  seconds seconds(aSecond);
  std::cout << seconds.count() << " sec\n";         // 1 sec

  minutes minutes(duration_cast<minutes>(aSecond));
  std::cout << minutes.count() << " min\n";         // 0 min

  typedef std::chrono::duration<double, std::ratio<2700>> MyLessonTick;
  MyLessonTick myLesson(aSecond);
  std::cout << myLesson.count() << " less\n";       // 0.00037037 less

  return 0;
}
```



Durations

i std::ratio

std::ratio supports arithmetic at compile time with rational numbers. A rational number has two template arguments. One is the nominator, the other the denominator. C++11 predefines lots of rational numbers.

```
typedef ratio <1, 1000000000000000000> atto;
typedef ratio <1, 1000000000000000> femto;
typedef ratio <1, 1000000000000> pico;
typedef ratio <1, 1000000000> nano;
typedef ratio <1, 1000000> micro;
typedef ratio <1, 1000> milli;
typedef ratio <1, 100> centi;
typedef ratio <1, 10> deci;
typedef ratio < 10, 1> deca;
typedef ratio < 100, 1> hecto;
typedef ratio < 1000, 1> kilo;
typedef ratio < 1000000, 1> mega;
typedef ratio < 1000000000, 1> giga;
typedef ratio < 1000000000000, 1> tera;
typedef ratio < 1000000000000000, 1> peta;
```

```
typedef ratio < 1000000000000000000, 1> exa;
```

C++14 has built-in literals for the most used time durations.

Type	Suffix	Example
`std::chrono::hours`	h	5h
`std::chrono::minutes`	min	5min
`std::chrono::seconds`	s	5s
`std::chrono::milliseconds`	ms	5ms
`std::chrono::microseconds`	us	5us
`std::chrono::nanoseconds`	ns	5ns

Built-in literals for time durations