CS253 HW7: Iteration!

Description

For this assignment, you will build upon your previous work in HW5.

Your **`Schedule`** class now works with the *for (auto v : container)* syntax! This implies support for **`Schedule::begin()`**, **`Schedule::end()`**, and **`Schedule::iterator`**. Looping over a **`Schedule`** produces read-only references to **`Events`**, in chronological order.

Methods

New **`Schedule`** features

The following methods & operators must work, where *sit* is of type **`Schedule::iterator`**.

`Schedule::begin()`
: Returns a value of type **`Schedule::iterator`** that corresponds to the first **`Event`** in the **`Schedule`**.

`Schedule::end()`
: Returns a value of type **`Schedule::iterator`** that corresponds to one *past* the last **`Event`** in the **`Schedule`**. Past, I say! It does **not** correspond to the last item, since **`begin()`** & **`end()`** form a half-open interval.

++sit
sit++
--sit
sit--
: Increments or decrements the iterator. Preincrement/predecrement return the new value, and postincrement/postdecrement return the previous value, in the same manner as `++` and `--` work on integers.

`*`*sit*
: Yields a **`const`** reference to the **`Event`** associated with the iterator.

sit == *sit*
sit != *sit*
: Compares two iterators for equality or inequality. Any other comparisons are undefined operations.

copy, assignment
: Iterators are copy-constructable, and assignable.

New **`Event`** features

`.fmt(`string **`format)`**
: Format the date according to the optional **`strftime()`** *format*, and return the resultant **`string`**. If *format* is not given, format in 10-character YYYY-MM-DD style. This does *not* change the default output format—it only affects the result of this method.

`Event++`
`++Event`
`Event--`
`--Event`
: Change this **`Event`** to refer to the next or previous day. Throw a **`runtime_error`** if this results in an out-of-range date (year 0 or 10000). Incrementing Dec 31 results in Jan 1 of the next year, and decrementing behaves correspondingly. Preincrement/decrement returns the new value, whereas postincrement/decrement returns the previous value.

`Event == Event`
`Event != Event`
`Event <= Event`
`Event >= Event`
`Event < Event`
`Event > Event`
: Compare two events. Two events are equal if they occur on the same date. One event is less than another if it occurs earlier.

Const-correctness is your job for all methods & operators.

Lifetime

Altering a **`Schedule`** potentially invalidates the corresponding iterators.

Sample Run

This focuses on the features added in this assignment. This does *not* imply that the previous features are abandoned.

```
% cat CMakeLists.txt
cmake_minimum_required(VERSION 3.14)
project(Homework)

# Using -Wall is required:
add_compile_options(-Wall)

# These compile flags are highly recommended, but not required:
add_compile_options(-Wextra -Wpedantic)

# Optional super-strict mode:
add_compile_options(-fmessage-length=80 -fno-diagnostics-show-option)
add_compile_options(-fstack-protector-all -g -O3 -std=c++14 -Walloc-zero)
add_compile_options(-Walloca -Wctor-dtor-privacy -Wduplicated-cond)
add_compile_options(-Wduplicated-branches -Werror -Wfatal-errors -Winit-self)
add_compile_options(-Wlogical-op -Wold-style-cast -Wshadow)
add_compile_options(-Wunused-const-variable=1 -Wzero-as-null-pointer-constant)

# add_compile_options must be BEFORE add_executable or add_library.

add_library(hw7 Event.cc Schedule.cc translate.cc)
add_executable(test test.cc)
target_link_libraries(test hw7)

# Create a tar file every time:
add_custom_target(hw7.tar ALL COMMAND tar cf hw7.tar Event.cc Event.h Schedule.cc Schedule.h translate.cc test.cc CMakeLists.txt)
% cat test.cc
#include "Schedule.h"
#include "Event.h"
#include "Schedule.h"       // I meant to do that.
#include "Event.h"
#include <iostream>
#include <sstream>
#include <cassert>

using namespace std;

int main() {
    istringstream tyt("tomorrow yesterday today");
    Schedule s(tyt);
    istringstream more("\t\r0100-10-10\f3000.365  ");
    s.read(more);
    // Should now contain:
    // 0: 0100-10-10
    // 1: yesterday
    // 2: today
    // 3: tomorrow
    // 4: 3000-12-31
    const auto s2(s);
    assert(s.size() == s2.size());
    for (size_t i=0; i<s2.size(); i++)
        assert(s[i] == s2[i]);
    s.clear();
    assert(s.empty());
    assert(s.size() == 0);
    for (const Event &e : s2)
        cout << e.fmt() << e.fmt(" / %04Y.%j / %A %B %e %04Y%n");

    Schedule::iterator it = s2.begin();
    assert(+++it == s2[1]);
    assert(*it++ == s2[1]);
    assert(*it++ == s2[2]);
    assert(*it++ == s2[3]);
    assert(*it++ == s2[4]);
    assert(it == s2.end());
    assert(it-- == s2.end());
    assert(*it-- == s2[4]);
    assert(*--it == s2[2]);

    auto yesterday = *--it;
    auto today = *++it;
    auto tomorrow = *++it;

    assert(yesterday == s2[1]);
    assert(today == s2[2]);
    assert(tomorrow == s2[3]);

    const Event &first = s2[0];
    assert(first < today);
    assert(first <= today);
    assert(first != today);
    assert(today > first);
    assert(today >= first);

    return 0;
}
% cmake .
… cmake output appears here …
% make
… make output appears here …
% ./test
0100-10-10 / 0100.283 / Sunday October 10 0100
2020-05-10 / 2020.131 / Sunday May 10 2020
2020-05-11 / 2020.132 / Monday May 11 2020
2020-05-12 / 2020.133 / Wednesday May 12 2020
3000-12-31 / 3000.365 / Wednesday December 31 3000
```

Requirements

Same as HW5, plus the additional features above, and:

- The result of an invalid **`strftime()`** format is undefined.
- You may assume that the **`strftime()`** format string will result in no more than 64 characters.
- Indirection on an end iterator produces undefined behavior.
- Incrementing an end iterator, or decrementing a begin iterator, produces undefined behavior.

Tar file

- The tar file for this assignment must be called: **`hw7.tar`**
- It must contain:
  - source files & header files needed to build your library
  - **`CMakeLists.txt`**, which will create the library file lib**`hw7.a`**.
- These commands must produce the library lib**`hw7.a`**:
  `cmake . && make`
- Your **`CMakeLists.txt`** must use *at least* **`-Wall`** when compiling.

How to submit your homework:

- Use web checkin, or Linux checkin:

```
~cs253/bin/checkin HW7 hw7.tar
```

How to receive *negative* points:

Turn in someone else's work.