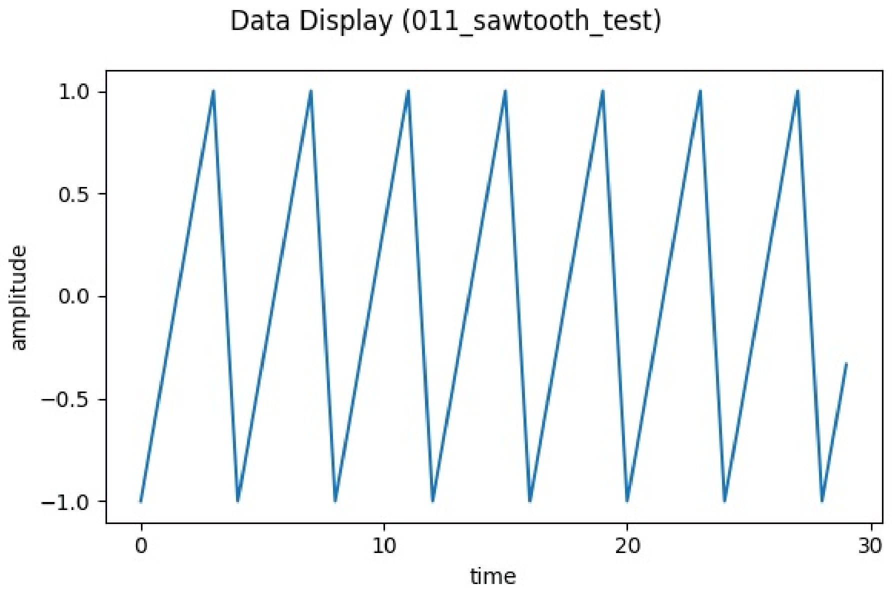
Sawtooth Waveform

Introduction

This task will add an additional waveform type to the library. All commands that use the waveform factory will be able to create this new waveform type. The waveform type will be called `sawtooth`. The sawtooth ramps from -maximum to +maximum in a cycle.



The pseudo code for the algorithm to use for computing the value looks like this:

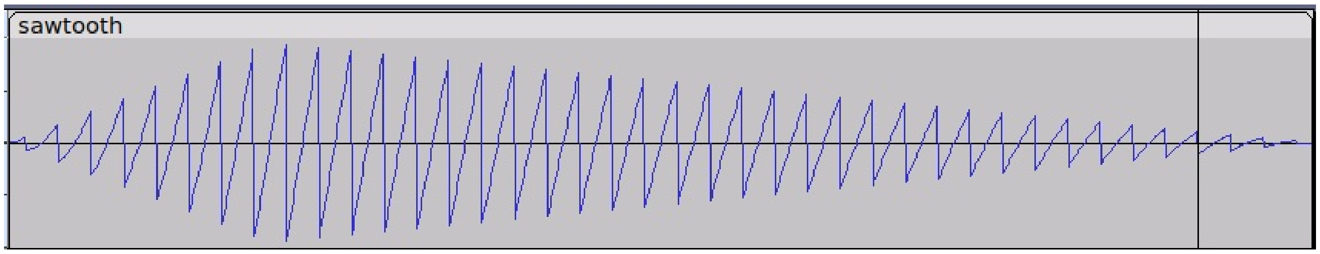
```
if cycle_position < 0.5
    value = 2 * cycle_position * maximum_amplitude
else
    value = (2 * (cycle_position - 0.5) - 1.0) * maximum_amplitude
```

Where `cycle_position` can be computed using the method of the `Waveform` class. This class implementation will look very similar to the `SquareWaveform` class from the assignment.

Example Usage

```
$ ./program-instrument-designer/instrument_designer
Choice? add-waveform
Waveform name: wf
Waveform type: sawtooth
Amplitude: 1.0
Choice? add-envelope
Envelope name: AD
Envelope type: AD
Maximum amplitude: 1.0
Attack seconds: 0.2
Choice? add-instrument
Instrument name: wfad
Waveform name: wf
Envelope name: AD
Choice? configure-audio-track-and-wav-file
Samples/Second: 1000
Seconds: 1.0
Bits/Sample[8,16,24,32]: 16
Choice? record-instrument-note
Instrument name: wfad
Frequency: 40.0
WAV filename: sawtooth.wav
Choice? quit
```

Opening the WAV file with Audacity or other audio editor should show the sawtooth waveform.



Programming Requirements

Some files already exist from the homework. Add to them to complete this task. Other files should be created as needed.

Create `library-waveform/SawtoothWaveform.{h,cpp}`

`SawtoothWaveform` Class

Publicly inherits from `Waveform`.

Data Members:

The `SawtoothWaveform` class does not define any new data members.

`public` Methods:

- `SawtoothWaveform(const std::string& name);` Constructor chains to the base class constructor. Uses "sawtooth" for the type name.
- `virtual ~SawtoothWaveform();` Empty body, but required.
- `double generateOneSample(const double frequency, const int sample_number, const double samples_per_second) const override;` Computes the value for a single sample. The algorithm is described above.

Update `library-waveform/WaveformFactory.{h,cpp}`

`public` Class Data Members:

- `const static std::vector<std::string> WaveformName;` Add "sawtooth" to the list of waveform names.

`public` Enumerations:

- `WaveformId` Add `WF_SAWTOOTH` to the enumeration.

`public` Methods:

- `static std::unique_ptr<Waveform> create(WaveformId id, const std::string& name);` Update to have handle the sawtooth case.

Grading Instructions

To receive credit for this task:

- Your code must be pushed to your repository for this class on GitHub.
- All unit tests for assignments and this task must pass.
- All acceptance tests for assignments must pass.
- All programs must build, run, and execute as described in the assignment descriptions.