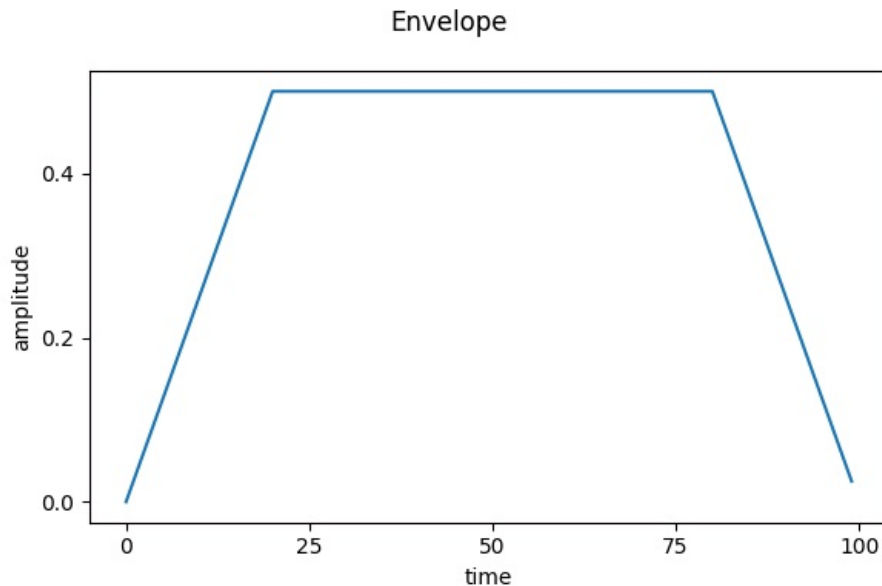


AR Envelope

Introduction

This task will add an additional envelope type to the library. All commands that use the envelope factory will be able to create this new envelope type. You will also update the commands that add and edit envelopes to configure the new envelope type.

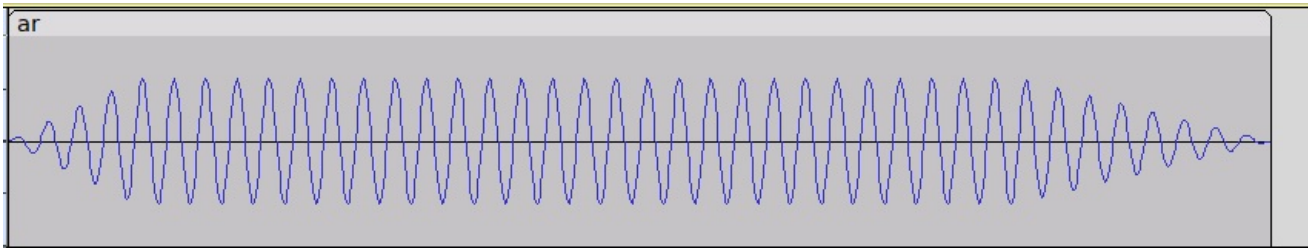
The waveform type will be called `AR`. The AR envelope is a subclass of the ADSR type envelope that has attack, sustain, and release phases, but no decay phase. The attack phase ramps from 0.0 to sustain amplitude. The release phase ramps from sustain amplitude to 0.0.



Example Usage

```
$ ./program-instrument-designer/instrument_designer
Choice? add-waveform
Waveform name: s
Waveform type: sine
Amplitude: 1.0
Choice? add-envelope
Envelope name: ar
Envelope type: AR
Maximum amplitude: 1.0
Attack seconds: 0.1
Sustain amplitude: 0.6
Release seconds: 0.2
Choice? add-instrument
Instrument name: sar
Waveform name: s
Envelope name: ar
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: sar
Frequency: 40.0
WAV filename: ar.wav
Choice? quit
```

Opening the WAV file with Audacity or other audio editor should show the AR envelope.



Note that the AR envelope needs the attributes `Maximum amplitude`, `Attack seconds`, `Sustain amplitude`, and `Release seconds`. This will be important when you add or edit an AR envelope.

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-envelope/AREnvelope.{h,cpp}`

`AREnvelope` **Class**

Publicly inherits from `ADSREnvelope`.

Data Members:

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

`public` Methods:

- `AREnvelope(const std::string& name);` Passes the first parameter to the parent class via constructor chaining. Also passes the values “AR”, 1.0, 0.0, 0.0, 0.0, and 0.0.
- `AREnvelope(const std::string& name, const double maximum_amplitude, const double attack_seconds, const double sustain_amplitude, const double release_seconds);` Constructor chains to the parent class, using the parameters here. Also uses “AR” for the type name, and 0.0 for all other unspecified values.
- `virtual ~AREnvelope();` Empty body, but required.
- `virtual void generateAmplitudes(const double seconds, const int samples_per_second, AudioTrack& track) const;` Generates amplitudes for the AR envelope, as described above.

Update `library-envelope/EnvelopeFactory.{h,cpp}`

`public` Class Data Members:

- `const static std::vector<std::string> EnvelopeName;` Add “AR” to the list of names.

`public` Enumerations:

- `EnvelopeId` Add `EN_AR` to the enumeration.

`public` Methods:

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

Update `library-commands/instrument_designer_aux.{h,cpp}`

Functions:

- `void addEnvelopeUI(ApplicationData& app_data);` Update to handle the AR case.
- `void editEnvelopeUI(ApplicationData& app_data);` Update to handle the AR 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.