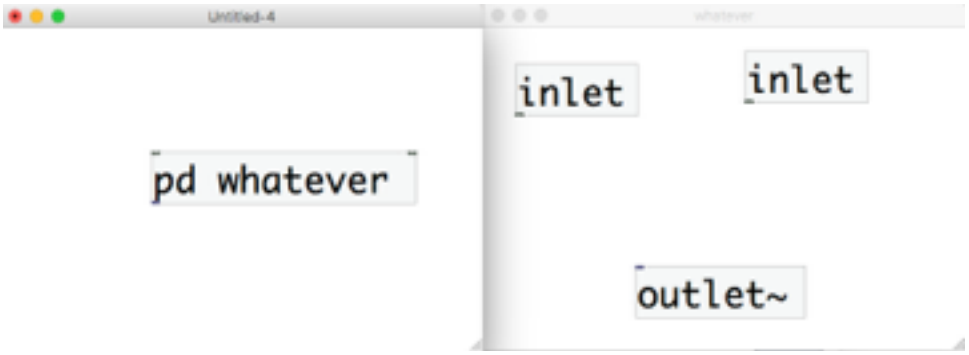
Tutorial – Week 4

1. Open **wk4-ex10-triangle-wave-spectrum.pd** located in the Examples folder.

   Here, we used the formula ***f*(x) = 2/π*arcsin(sin(x))** to create a triangle wave whose amplitude range is -1.0 to 1.0. Can you think of another, simpler way of doing this, starting with the ramp from 0.0 to 1.0 that the `phasor~` object outputs? (hint: the first half of the ramp is already the rising half of a triangle wave—you need the second half of the ramp to go down rather than up! Remember you may need to scale the output to fit into the range -1.0 to 1.0).

2. A subpatch is a Pd structure that allows you to encapsulate certain parts of your patch. Create a new object and type in `pd whatever`. A new blank window will appear (named 'whatever'). Within the new window, create a couple of `inlet` objects and one `outlet~`. You will notice on your original patch, that `pd whatever` now has two control rate inlets and a signal rate outlet. Try experimenting with more inlets and outlets. By creating subpatches you can modularize certain processes of your patch, in order to make it easier to manage when it gets messy (We'll look further into subpatches, abstractions and externals in future lectures).



3. Create a subpatch `pd triangle` for the triangle wave oscillator. The first inlet should be the frequency of the oscillator in Hz, the second inlet the amplitude of the output. Create on outlet for the oscillator output. Remember use the `line~` object as in the original patch to smooth the amplitude data. Test this subpatch by inputting frequency and amplitude data from sliders and sending the output to the `dac~` object.

4. Using the same principle and `phasor~` as your starting point, create similar subpatches for sine, square, sawtooth. (Make sure you scale their outputs from -1.0 to 1.0 where required)

5. Challenge: Create a pulse wave generator, a general version of the square wave where you control the width of the 1s and -1s. You will need an extra inlet to control the pulse width.