Problem Wk.4.1.1: Constructing Signals

Enter Python expressions to construct the following signals, using the signal constructors defined in the software lab handout. **Read the entire software lab handout before beginning this problem.**

The signal constructors are already defined in this problem. **You should not define any new Python classes.**

We recommend that you try this in Idle first, using the software that we have provided with the software lab. If you add Python expressions to the `swLab04AWork.py` file, you have access to the pre-implemented Signal classes and constructors: ConstantSignal, UnitSampleSignal, CosineSignal, StepSignal, SummedSignal, ScaledSignal, R, Rn and polyR (all defined in the handout).

When you paste your code from Idle into the tutor, do not include the `import` statements.

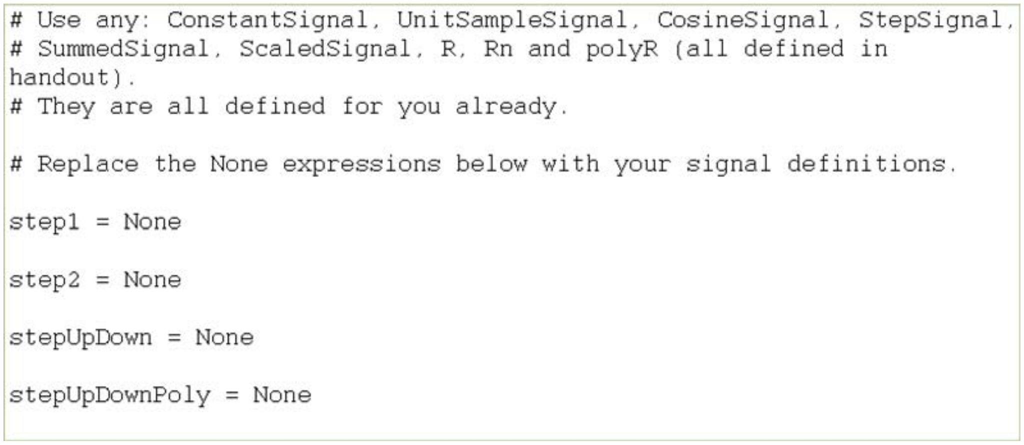
Make sure that you use the correct variable name for each example.

- `step1`: this signal should be equal to 3.0 for any time $t \geq 3$ and should equal 0 otherwise.
- `step2`: this signal should be equal to -3.0 for any time $t \geq 7$ and should equal 0 otherwise.
- `stepUpDown`: this signal should be equal to 3.0 for any time $3 \leq t \leq 6$ and should equal 0 otherwise. Hint: Can you use `step1` and/or `step2`?
- `stepUpDownPoly`: Use the `polyR` class to construct a signal that has value 1.0 at time 1, value 3.0 at time 3, value 5.0 at time 5 and is 0 everywhere else.

The testing for this problem and subsequent ones will use the following function to construct a list of samples to compare:

```
def samplesInRange(signal, lo, hi):
    return [signal.sample(i) for i in range(lo,hi)]
```

If you want to use `polyR`, notice that the second argument to `polyR` is an instance of the `Polynomial` class. You can create new instances via `poly.Polynomial(c)` where c is a list of coefficients.



```
# Use any: ConstantSignal, UnitSampleSignal, CosineSignal, StepSignal,
# SummedSignal, ScaledSignal, R, Rn and polyR (all defined in
handout).
# They are all defined for you already.

# Replace the None expressions below with your signal definitions.

step1 = None

step2 = None

stepUpDown = None

stepUpDownPoly = None
```

6.01SC Introduction to Electrical Engineering and Computer Science
Spring 2011