Homework 6

March 5, 2020

1 Introduction

For this homework, we will reuse all definitions in the last homework. You will add a function `makeNoise` to generate a Vec of random values within a range.

```
makeNoise :: Random a => Int -> Int -> a -> a -> Vec [a]
```

where `makeNoise seed n low high` returns a Vec of `n` random numbers between `low` and `high` with the random `seed`. For example, the call below returns 10 random numbers between 1 and 6 with seed 0.

```
makeNoise 0 10 1 6
-- [6 6 4 1 5 2 4 2 2 1]
```

The call below returns 10 random numbers between 0.0 and 1.0 with seed 12345.

```
rd 1 $ makeNoise 12345 10 0.0 1.0
-- [0.2 0.7 0.9 1.0 0.8 0.3 0.7 0.3 0.3 0.7]
```

You should import `System.Random` module, where the class `Random` has a function `randomR :: (Random a, RandomGen g) => (a, a) -> g -> (a, g)` that takes a `(low, high)` pair and a random number `generator` and returns a `(random number, generator)` pair.

You can use `mkStdGen :: Int -> StdGen` to create a generator from an Int seed. Since `StdGen` is an instance of `RandomGen`, we can use `mkStdGen` to create an initial generator for `randomR`. Each time you use `randomR`, a new generator is returned, which should be used to generate the next random number.

2 Requirement

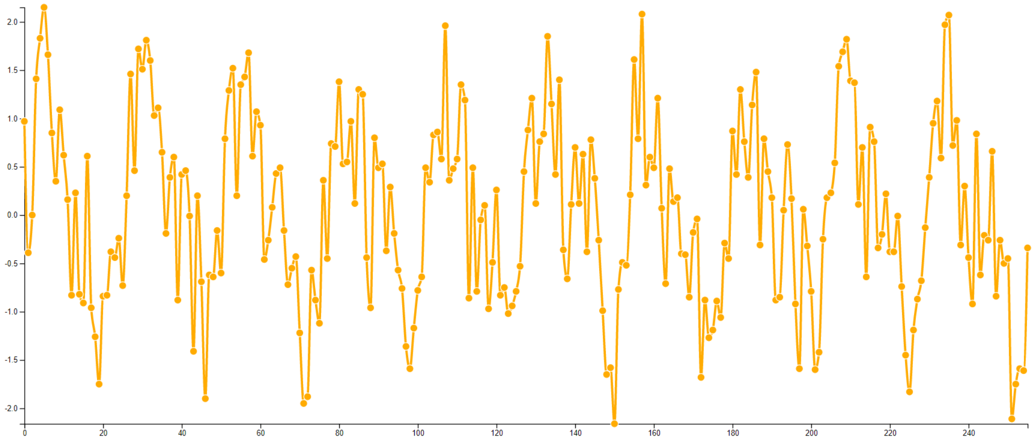
You should use State Monad to keep track of the generator state.

3 Testing

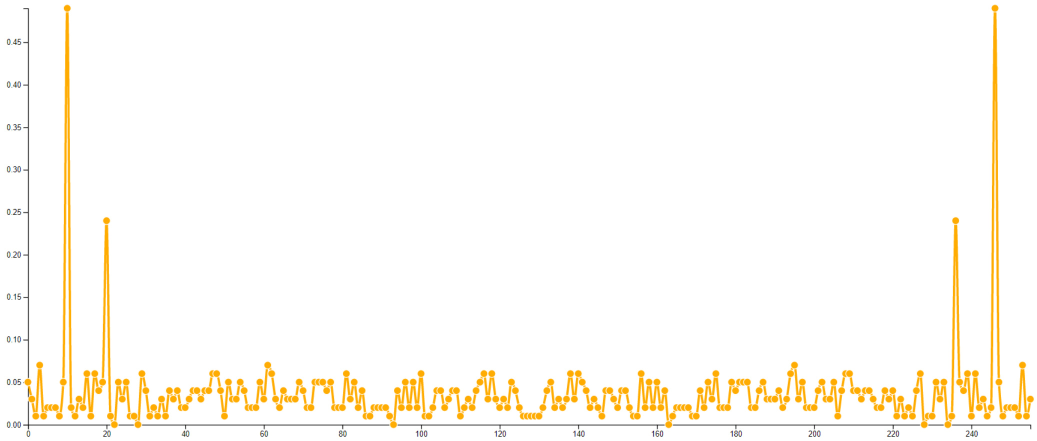
You can test the implementation using the following `main`

```
main = do
         let n’ = 2^8
         let noise = makeNoise 0 n’ (-1.0) 1.0
         let n = fromIntegral n’
         let s = fmap (\x -> sin(20*pi*x) + sin(40*pi*x)/2) $ Vec $ range 0 1 n
         let s1 = s + noise

         -- noisy signal
         print(rd 2 s1)

         -- noisy frequency
         print(rd 2 $ fmap (/n) $ absolute $ dft $ realV s1)

         -- filtered signal (dft)
         print(rd 3 $ fmap (\(r:+_) -> r) $ low_pass’ 15 $ realV s1)

         -- filtered signal (fft)
         print(rd 2 $ fmap (\(r:+_) -> r) $ low_pass 15 $ realV s1)
```

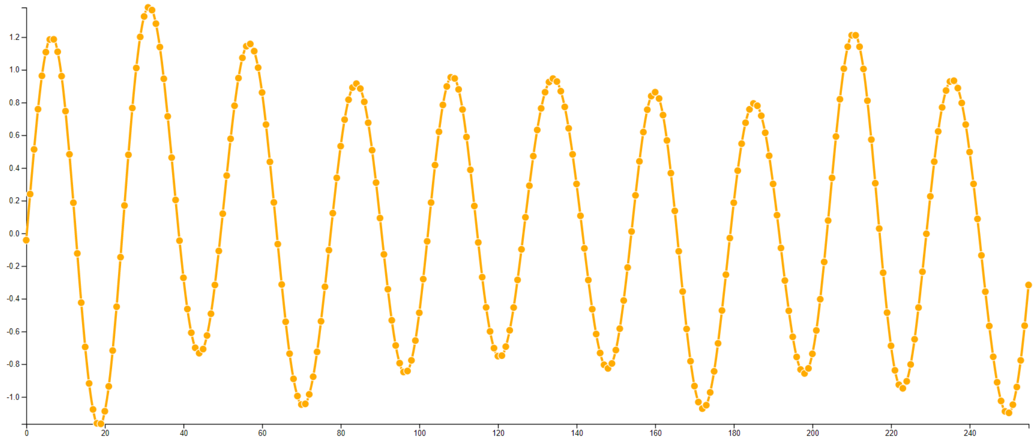
You can use the HTML file I provide in last homework to visualize the signal output. The signal with noise looks like the following.



The noisy signal after DFT looks like the following.



The filtered signal looks like the following.



4 Submission

Please write your solution in a file – `hwk6.hs` and submit it to the dropbox.