BEE 4750/5750 Homework 0

Katerina Tang (kbt28)

2022-08-29

Problem 1

Problem 1.1

```
julia> function square_number(x)
           output = x.^2
           return output
       end
square_number (generic function with 1 method)
```

Problem 1.2

If $x = 5$, then $x^2 = 25$.

Problem 1.3

```
julia> x = LinRange(-10, 10, 100)
100-element LinRange{Float64}:
 -10.0,-9.79798,-9.59596,-9.39394,…,9.19192,9.39394,9.59596,9.79798,10.0

julia> y = square_number(x)
100-element Vector{Float64}:
 100.0
  96.00040812162027
  92.08244056728904
  88.24609733700643
  84.49137843077234
  80.81828384858687
  77.22681359044998
  73.71696765636158
  70.2887460463218
  66.94214876033055

  70.2887460463218
  73.71696765636158
  77.22681359044998
```

```
  80.81828384858687
  84.49137843077234
  88.24609733700643
  92.08244056728904
  96.00040812162027
 100.0

julia> plot(x, y, xlabel="x", ylabel="y = x^2", legend=false)
Error: UndefVarError: plot not defined
```

Problem 2

Problem 2.1

If $a \leq \sqrt{x}$, then

$$\frac{x}{a} \geq \frac{x}{\sqrt{x}} = \sqrt{x}.$$

If $a > \sqrt{x}$, then

$$\frac{x}{a} < \frac{x}{\sqrt{x}} = \sqrt{x}.$$

In both cases, $a \leq \sqrt{x} \leq \frac{x}{a}$.

Problem 2.2

```
julia> function newton_sqrt(x, err_tol)
           a = x

           while true
               root = 0.5 * (a + (x/a))
               if abs(root - a) < err_tol
                   return root
               end
               a = root
           end
       end
newton_sqrt (generic function with 1 method)
```

Using this method and an error tolerance of 10^{-8} gives $\sqrt{2} \approx 1.414213562373095$.

Problem 3

Problem 3.1

```
julia> a = rand(20)
20-element Vector{Float64}:
 0.4202808246502596
 0.21866027687081324
 0.003376276250591337
 0.12762843233067112
 0.5233495533263068
 0.5710618331437627
 0.5343428085604245
 0.833173099324066
 0.12424693420564736
 0.15866442240078849
 0.8856584024327951
 0.9845063904954752
 0.8288470946700346
 0.25662895092713933
 0.10621458147938845
 0.396527307370085
 0.840113295539646
 0.9221100735868546
 0.8036892400034941
 0.9998473996654129
```

Problem 3.2

```
julia> function mean(vect)
           sum = 0

           for i in 1:length(vect)
               sum += vect[i]
           end

           return sum/length(vect)
       end
mean (generic function with 1 method)

julia> function demean(vect)
           return vect .- mean(vect)
       end
demean (generic function with 1 method)
```

We can test this with our random vector from part (1).

```
julia> mean(a)
0.5269463598616828
```

```
julia> demean(a)
20-element Vector{Float64}:
 -0.10666553521142319
 -0.30828608299086957
 -0.5235700836110915
 -0.3993179275310117
 -0.0035968065353759737
  0.044115473282079876
  0.0073964486987416755
  0.3062267394623832
 -0.40269942565603545
 -0.3682819374608943
  0.35871204257111233
  0.45756003063379236
  0.3019007348083518
 -0.2703174089345435
 -0.42073177838229436
 -0.1304190524915978
  0.31316693567796317
  0.3951637137251718
  0.27674288014181125
  0.4729010398037301
```

Problem 3.3

```
julia> b = zeros(10);

julia> b[3:8] .= 1.0;

julia> b
10-element Vector{Float64}:
 0.0
 0.0
 1.0
 1.0
 1.0
 1.0
 1.0
 1.0
 0.0
 0.0
```

Problem 3.4

```
julia> A = rand(5, 5)
5×5 Matrix{Float64}:
 0.96482   0.579579  0.0331459  0.562332   0.558857
 0.741152  0.182883  0.310079   0.752733   0.693944
```

```
 0.790639  0.585683  0.140372   0.462211   0.351487
 0.569175  0.76409   0.999067   0.0700408  0.887837
 0.829565  0.904767  0.199736   0.790991   0.944955
```

```
julia> for i in 1:5
           A[:, i] .-= mean(A[:, i])
       end

julia> A
5×5 Matrix{Float64}:
  0.18575    -0.0238216  -0.303334    0.0346702  -0.128559
 -0.0379182  -0.420518   -0.0264007   0.225071    0.00652799
  0.0115691  -0.0177178  -0.196108   -0.0654501  -0.335929
 -0.209895    0.16069     0.662587   -0.457621    0.200421
  0.0504945   0.301367   -0.136744    0.263329    0.257539
```

Problem 4

Problem 4.1

```
julia> d = LogNormal(log(0.03), 0.005)
Error: UndefVarError: LogNormal not defined

julia> y = rand(d, 100)
Error: UndefVarError: d not defined
```

Problem 4.2

```
julia> function P_levels(a, y, b, q, T, X0)
           X = zeros(T)
           X[1] = X0

           for i in 2:T
               X[i] = X[i-1] + a + y[i] + X[i-1]^q / (1+ X[i-1]^q) - b*X[i-1]
           end

           return X
       end
P_levels (generic function with 1 method)
```

Problem 4.3

```
X = P_levels(0.4, y, 0.42, 2, 100, 1)
plot(X, xlabel = "time (years)", ylabel="phosphorous level", legend = false)
```

References