Vector & Julia Basics

This is the first project of the course

It will include the basic of vectors and Julia There are 5 question regarding this topics:

1. Inner Product
2. Exception Handling
3. Advanced Exception Handling
4. Eugene Calculator
5. Sunny's Crazy Idea

1. Inner Product

(Inner Product) Define a function called "Inner_Product" that produces the result of the inner product of two vectors.

- Example: a = [1, 2, 3], b = [2, 3, 4], your function needs to output 20.
  Input: a, b
  Output: 20

```
function f(x,y)
          x'*y
       end
Inner_Product = f ;
```

Solution Description

the dot product formula = x' * y

2. Exception Handling

(Exception Handling) In the lecture, we have learnt that you cannot apply inner product to two vectors with different dimensions. Define a function called "Strict_inner_Product": if two vectors have the same dimension, output the result of their inner product; otherwise, output a warning and the dimension of two vectors.

- Example:

1. a = [1, 2, 3], b = [2, 3, 4, 5], your function should output "Warning! 3*1 vector can't do inner product with a 4*1 vector!"
   Input: a, b
   Output: Warning! 3*1 vector can't do inner product with a 4*1 vector!
2. a = [1, 2, 3], b = [2, 3, 4], your function should print 20. Input: a, b
   Output: 20

```
function g(x,y)
    if(size(x) != size(y))
        dx = size(x)
        dy = size(y)
        @warn  " $dx vector can't do inner product with a $dy vector!"
        return 0
    else
    x'*y
    end
```

```
end
Strict_inner_Product = g;
```

Solution Description

If statement (size z,y is different warn them)

3. Advanced Exception Handling

(Advanced Exception Handling) In the tutorial, TA taught how to identify the datatype of a variable. If there is a naughty student, named NS, who changes your variable, from an assigned vector to a string or a tuple, the previous functions won't work under NS's trick. Please save TA from NA's trick by creating a new function called "Identify_Wrong_Datatype" which will identity the datatype of your input, if they are allowed
to do inner product, then you output the result of the operation; otherwise, you output the datatype of input.

- Example:

1. a = [1, 2, 3], b = [2, 3, 4], your function should print 20. Input: a, b
   Output: 20
2. a = "[1, 2, 3]", b = [1, 2, 3], your function should output "Warning! String(datatype
   of a) can't do inner product with Vector(datatype of b)!" Input: a, b
   Output: Warning! String can't do inner product with Vector!

```
function h(x,y)
    if (typeof(x) != typeof(y))
        #println("Warning! 3*1 vector can't do inner product with a 4*1 vector!")
        tx = typeof(x)
        ty = typeof(y)
        @warn  " $tx can't do inner product with $ty !"
        return 0
    elseif (typeof(x) ==String &&  typeof(y)== String)
        tx = typeof(x)
        ty = typeof(y)
            @warn  " $tx can't do inner product with $ty !"
    else
    x'*y
    end
end
Identify_Wrong_Datatype = h;
```

Solution Description

If statement (datatype is different warn them) also I add String cant do Inner product with another String

4. Eugene Calculator

(Eugene's calculator) Eugene has a magic calculator. When you key in an integer (n) and the operands, it will randomly generate "n" numbers and output the result of mathematical operation with the input operands. The teaching team (TT) thinks it's so cool to have that calculator. As result, Eugene wants to give TT the magic calculator as a gift. Yet, TT can't reproduce the software inside the magic calculator.

- The logic of the calculator is as following:

1. Randomly generate n, the key-in integer, numbers from Normal distribution with
   mean equals to 0 and variance equals to 100.
2. Use the operands in the given vector in order to do the mathematical operation.

3. Output the result

Please help Eugene to develop the software and name it “Gift”.

- Example:
  Input: n = 4,
  operands = [‘+’, ‘-’, ‘*’] Output: 10 (suppose the generated number is a,b,c,d, and the result of (((a+b)- c)d) is 10)
  *Please don’t set random seed inside your function!

```
using Pkg
using Distributions
function Happy_Birthday(n,a)
    gd = Normal(0,10) #var is σ^2 so var = 100 can be written as σ = 10
    rrnd= map(x->round.(x), rand(gd,n,)) #rounding
    rrnd_int = convert(Vector{Int64}, rrnd) #convert to vector & integer
    println(rrnd_int)
    add = '+'
    sub = '-'
    mul = '*'
    div = '/'
    if length(a) > length(rrnd_int)
        @warn  " too many operands"
    else
        if a[1] == add
            a1 = rrnd_int[1] + rrnd_int[2]
        elseif a[1] == sub
            a1 = rrnd_int[1] - rrnd_int[2]
        elseif a[1] == mul
            a1 = rrnd_int[1] * rrnd_int[2]
        elseif a[1] == div
            a1 = rrnd_int[1] / rrnd_int[2]
        else
        println("too long I dont know what loop to choose yet")
    end
    end
end
```

Solution Description

First I randomly generate n, the key-in integer, numbers from Normal distribution with mean equals to 0 and variance equals to 100
(p.s. I didnt use randn because I cant set the mean and variance the standard is 0,1 but here we want in 0,100 since in the function Normal I can declare (mean,sd) since variance is sd^2 so i set the sd =10 10^2=100)
Then I set the input and use if else function to build the calculator
I haven't figured out how to loop it (at that time)

5. Sunny’s Crazy Idea

(Sunny’s Crazy Idea) When Sunny claims reimbursement, sometimes he finds there are some missing values on his sheet. With those missing values, he can’t get the total expense he should receive easily by doing inner product. To avoid errors, he asks his administrative assistant (AA) to create a new function called “Account_Manager”, which will automatically fill the missing value by replicating the value(s) from the top of the vector. Moreover, to be careful, you also need to tell Sunny which value(s) is/are missing and what value do you use to substitute the missing value. The information he has are the names of equipment he bought, their quantities and their prices.

- Example:

1. name = [‘Pencil’, ‘Marker’, ‘Glue’], quantity = [3, 4], price = [30, 50, 80]. The total
   expense will be 3*30* + *4*50 + 3*30 = 530. Then you should print “The total expense is 530. The quantity for Glue is missing and filled with 3.”

Input: name, quantity, price
Output: The total expense is 530.
The quantity for Glue is missing and filled with 3.

2. name = [‘Pencil’, ‘Marker’, ‘Glue”], quantity = [3, 4, 5], price = [30, 50]. The total expense will be 3*30* + 4*50* + 5*30 = 440. Then you should print “The total expense is 440. The price for Glue is missing and filled with 30.”
   Input: name, quantity, price
   Output: The total expense is 440.
   The price for Glue is missing and filled with 30.
3. name = [‘Pencil’, ‘Marker’, ‘Glue’, ‘Scissor’], quantity = [3, 4], price = [30, 50, 80].
   3*30* + 4*50* + 3*80* + 4*30* = 650. Then you should print “The total expense is 650. The quantity for Glue is missing and filled with 3. The quantity for Scissor is missing and filled with 4. The price for Scissor is missing and filled with 30.”
   Input: name, quantity, price
   Output: The total expense is 650.
   The quantity for Glue is missing and filled with 3. The quantity for Scissor is missing and filled with 4. The price for Scissor is missing and filled with 30.

```
function Account_Manager(n,q,p)
    if length(q) < length(p)
        y = length(p) - length(q)
        gd = Normal(0,10) #var is σ^2 so var = 100 can be written as σ = 10
        rrnd= map(x->round.(x), rand(gd,y,)) #rounding
        rrnd_int = convert(Vector{Int64}, rrnd)  #convert to vector & integer and back to previous
        println(rrnd_int)
        println("ignore negative I've square it  ")
        plast = last(p)
        p2 =setdiff(p, plast)
        b = (q' * p2 ) #dot product without the NA
        c = ((rrnd_int)[1]) # convert into scalar
        c_sq = c^2 #remove negative
        c_sqrt = sqrt(c_sq) #back to previous value
        c1 = c_sqrt * plast #remanding missing value dot product
        d = b + c1 #remanding missing value dot product
    elseif length(q) > length(p)
        y = length(q) - length(p)
        gd = Normal(0,10) #var is σ^2 so var = 100 can be written as σ = 10
        rrnd= map(x->round.(x), rand(gd,y,)) #rounding
        rrnd_int = convert(Vector{Int64} , rrnd) #convert to vector & integer
        println(rrnd_int)
        println("ignore negative I've square it  ")
        qlast = last(q)
        q2 =setdiff(q, qlast) #because there is repeat of 1 in q 4 it didnt work
        b = (q2' * p ) #dot product without the NA
        c = ((rrnd_int)[1]) # convert into scalar
        c_sq = c^2 #remove negative
        c_sqrt = sqrt(c_sq)
        c1 = c_sqrt * qlast
        d = b + c1
    else
        b = q' * p
    println(b)
    end
end
```

Solution Description

At first I thought it was a “reverse”dot product but as I saw the test from what I understand we didn't have the total yet and we have na that have to be substitute so

- First make a if statement to know which department is missing quantity or quality
- Then make a random integer value that I took from question 3. And square it and sqrt it to remove the negative.
- Then I remove the last integer from the the “longer” vector and do dot product

- then I add the random value * the last scalar and add it to the dot product
  (p.s. In the random number I square it and then square root it to make it positive)