MLBA homework1

In [6]:

```
# Put the package you need to use here
using Random
```

Q1 Inner Product

Define a function on the below block, and name it "Inner_Product"

In [26]:

```
# Define you function on this block

function Inner_Product(a, b)
    sum = 0
    for i in 1:length(a)
        sum = sum + a[i]*b[i]
    end
    return sum
end
```

Out[26]:

```
Inner_Product (generic function with 1 method)
```

Run the below blocks to get marks

In [27]:

```
# Q1 Test 1
Q1_1_v1 = [1, 2, 3]
Q1_1_v2 = [1, 2, 5]
println(Inner_Product(Q1_1_v1, Q1_1_v2))
```

```
20
```

In [28]:

```
# Q1 Test 2
Q1_2_v1 = [1.82, -2.56, 3.64]
Q1_2_v2 = [-1.43, -0.788, 5.3829]
println(Inner_Product(Q1_2_v1, Q1_2_v2))
```

```
19.008436000000003
```

In [29]:

```
#Q1 Test 3
Q1_3_v1 = [1 3 5]
Q1_3_v2 = [1, 3, 4]
println(Inner_Product(Q1_3_v1, Q1_3_v2))
```

```
30
```

In [30]:

```
#Q1 Test 4
Random.seed!(1314)
Q1_4_v1 = rand(10)
Q1_4_v2 = rand(10)
println(Inner_Product(Q1_4_v1, Q1_4_v2))
```

```
2.0411247582415086
```

```
2.0111217562115000
```

In [31]:

```
# Q1 Test 5
Random.seed!(9487)
Q1_5_v1 = rand(10000)*100
Q1_5_v2 = rand(10000)*100
println(Inner_Product(Q1_5_v1, Q1_5_v2))
```

```
2.5007532418357e7
```

Q2 Exception Handling

Define a function on the below block, and name it "Strict_inner_Product"

In [38]:

```
# Define your function on this block

function Strict_inner_Product(a, b)

    if length(a) == length(b)
        sum = 0
        for i in 1:length(a)
            sum = sum + a[i]*b[i]
        end
        return sum

    else
        len_a = length(a); len_b = length(b)
        return("Warning! $len_a*1 vector can’t do inner product with a $len_b*1 vector!"
)
    end

end
```

Out[38]:

```
Strict_inner_Product (generic function with 1 method)
```

Run the below blocks to get marks

In [39]:

```
# Q2 Test 1
Q2_1_v1 = [1, 2, 3]
Q2_1_v2 = [1, 2, 5]
println(Strict_inner_Product(Q2_1_v1, Q2_1_v2))
```

```
20
```

In [40]:

```
# Q2 Test 2
Q2_2_v1 = [1, 2, 3]
Q2_2_v2 = [1, 2, 5, 8]
println(Strict_inner_Product(Q2_2_v1, Q2_2_v2))
```

```
Warning! 3*1 vector can’t do inner product with a 4*1 vector!
```

In [41]:

```
# Q2 Test 3
Random.seed!(2468)
Q2_3_v1 = rand(1000)*100
Q2_3_v2 = rand(1000)*100
println(Strict_inner_Product(Q2_3_v1, Q2_3_v2))
```

```
2.50186549433591e6
```

In [42]:

```
# Q2 Test 4
Q2_4_v1 = [1 2 3 4 5]
Q2_4_v2 = [3, 4, 5, 6]
println(Strict_inner_Product(Q2_4_v1, Q2_4_v2))
```

```
Warning! 5*1 vector can’t do inner product with a 4*1 vector!
```

In [43]:

```
# Q2 Test 5
Q2_5_v1 = rand(100)
Q2_5_v2 = rand(1001)'
println(Strict_inner_Product(Q2_5_v1, Q2_5_v2))
```

```
Warning! 100*1 vector can’t do inner product with a 1001*1 vector!
```

Q3 Advanced Exception Handling

Define a function on the below block, and name it "Identify_Wrong_Datatype"

In [58]:

```
# Define your function on this block

function Identify_Wrong_Datatype(a, b)

    if a isa Vector && b isa Vector
            sum = 0
            for i in 1:length(a)
                sum = sum + a[i]*b[i]
            end
            return sum
    else
        type_a = typeof(a); type_b = typeof(b)
        return("Warning! $type_a(datatype of a) can’t do inner product with $type_b(datat
ype of b)!")
    end

end
```

Out[58]:

```
Identify_Wrong_Datatype (generic function with 1 method)
```

Run the below blocks to get marks

In [59]:

```
# Q3 Test 1
Q3_1_v1 = [1, 2, 3]
Q3_1_v2 = [1, 2, 5]
println(Identify_Wrong_Datatype(Q3_1_v1, Q3_1_v2))
```

```
20
```

In [60]:

```
# Q3 Test 2
Q3_2_v1 = [1, 2, 3]
Q3_2_v2 = "[1, 2, 5]"
println(Identify_Wrong_Datatype(Q3_2_v1, Q3_2_v2))
```

```
Warning! Vector{Int64}(datatype of a) can’t do inner product with String(datatype of b)!
```

In [61]:

```julia
# Q3 Test 3
Q3_3_v1 = "[1, 2, 3]"
Q3_3_v2 = [1, 2, 5]
println(Identify_Wrong_Datatype(Q3_3_v1, Q3_3_v2))
```

```
Warning! String(datatype of a) can’t do inner product with Vector{Int64}(datatype of b)!
```

In [62]:

```julia
# Q3 Test 4
Q3_4_v1 = "[1, 2, 3]"
Q3_4_v2 = "[1, 2, 5]"
println(Identify_Wrong_Datatype(Q3_4_v1, Q3_4_v2))
```

```
Warning! String(datatype of a) can’t do inner product with String(datatype of b)!
```

In [63]:

```julia
# Q3 Test 5
Q3_5_v1 = [1, 2, 3]
Q3_5_v2 = (1, 3, 4)
println(Identify_Wrong_Datatype(Q3_5_v1, Q3_5_v2))
```

```
Warning! Vector{Int64}(datatype of a) can’t do inner product with Tuple{Int64, Int64, Int
64}(datatype of b)!
```

Q4 Eugene’s calculator

Define a function on the below block, and name it "Happy_Birthday"

In [51]:

```julia
# Define the function on this block

function operation(a, b, c)
    num = 0
    if a == '+'
        num = b + c
    elseif a == '-'
        num = b - c
    elseif a == '*'
        num = b * c
    elseif a == '/'
        num = b / c
    end
    return num
end



function Happy_Birthday(a, b)
    r = randn(a)*100

    for i in 1:length(r)
        r[i] = round(r[i])
    end

    cal = 0
    for i in 1:length(b)
        if i == 1
            cal = operation(b[i], r[i], r[i+1])
        else
            cal = operation(b[i], cal, r[i+1])
        end
    end
    return cal
end
```

Out[51]:

```
Happy_Birthday (generic function with 1 method)
Happy_Birthday (generic function with 1 method)
```

Run the below blocks to get marks

In [52]:

```
# Q4 Test 1
Random.seed!(4129889)
Q4_1_integer = 2
Q4_1_operand = ['+']
println(Happy_Birthday(Q4_1_integer, Q4_1_operand))
```

```
-154.0
```

In [53]:

```
# Q4 Test 2
Random.seed!(800092000)
Q4_2_integer = 3
Q4_2_operand = ['+', '-']
println(Happy_Birthday(Q4_2_integer, Q4_2_operand))
```

```
-117.0
```

In [54]:

```
# Q4 Test 3
Random.seed!(870887)
Q4_3_integer = 4
Q4_3_operand = ['+', '-', '*']
println(Happy_Birthday(Q4_3_integer, Q4_3_operand))
```

```
18270.0
```

In [55]:

```
# Q4 Test 4
Random.seed!(7414666)
Q4_4_integer = 5
Q4_4_operand = ['+', '-', '*', '/']
println(Happy_Birthday(Q4_4_integer, Q4_4_operand))
```

```
1.509433962264151
```

In [56]:

```
# Q4 Test 5
Random.seed!(9481)
Q4_5_integer = 12
Q4_5_operand = ['+', '-', '*', '/', '/', '*', '+', '*', '-', '/', '+']
println(Happy_Birthday(Q4_5_integer, Q4_5_operand))
```

```
-247.31052631578947
```

Q5 Sunny's Crazy Idea

Define a function on the below block, and name it "Account_Manager"

In [15]:

```
# Define the function on this block

function Account_Manager(a, b, c)
    if length(a) == length(b) && length(b) == length(c)
        sum = b' * c
        return("The total expense is $sum.")
    else
        printout_list = "i"
        i = 1
        while length(a) > length(b)
            append!(b, b[i])
            printout_list = vcat(printout_list, "The quantity for $(a[length(b)]) is mis
sing and filled with $(b[length(b)]).")
            i = i + 1
        end

        j = 1
        missing_price = []
        while length(a) > length(c)
            append!(c, c[j])
            printout_list = vcat(printout_list, "The price for $(a[length(c)]) is missin
g and filled with $(c[length(c)]).")
            j = j + 1
        end

        sum = b' * c
        printout_list[1] = "The total expense is $sum."

        printout = ""

        for i in 1:length(printout_list)
            if i == 1
                printout = printout * printout_list[i]
            else
                printout = printout * "\n$(printout_list[i])"
            end
        end
        return(printout)
    end
end
```

Out[15]:

```
Account_Manager (generic function with 1 method)
```

Run the below blocks to get marks

In [16]:

```
# Q5 Test 1
Q5_1_name = ["Sunny", "Hsin", "Eric"]
Q5_1_quantity = [0, 1, 1]
Q5_1_price = [1, 10, 100]
println(Account_Manager(Q5_1_name, Q5_1_quantity, Q5_1_price))
```

```
The total expense is 110.
```

In [17]:

```
# Q5 Test 2
Q5_2_name = ["Sunny", "Hsin", "Eric", "Breakfast", "Dinner", "Concert"]
Q5_2_quantity = [0, 1, 1, 10, 20]
Q5_2_price = [1, 10, 100, 5, 50, 500]
println(Account_Manager(Q5_2_name, Q5_2_quantity, Q5_2_price))
```

```
The total expense is 1160.
The quantity for Concert is missing and filled with 0.
```

In [18]:

```
# Q5 Test 3
Q5_3_name = ["Sunny", "Hsin", "Eric", "Breakfast", "Dinner", "Concert"]
Q5_3_quantity = [0, 1, 1, 10, 20, 50]
Q5_3_price = [1, 10, 100, 5, 50]
println(Account_Manager(Q5_3_name, Q5_3_quantity, Q5_3_price))
```

```
The total expense is 1210.
The price for Concert is missing and filled with 1.
```

In [19]:

In [19]:

```
# Q5 Test 4
Q5_4_name = ["Sunny", "Hsin", "Eric", "Breakfast", "Dinner", "Concert"]
Q5_4_quantity = [0, 1, 1, 10, 20]
Q5_4_price = [1, 10, 100, 5, 50]
println(Account_Manager(Q5_4_name, Q5_4_quantity, Q5_4_price))
```

```
The total expense is 1160.
The quantity for Concert is missing and filled with 0.
The price for Concert is missing and filled with 1.
```

In [20]:

```
# Q5 Test 5
Q5_5_name = ["Sunny", "Hsin", "Eric", "Breakfast", "Dinner", "Concert"]
Q5_5_quantity = [0, 1, 1, 10]
Q5_5_price = [1, 10, 100, 5]
println(Account_Manager(Q5_5_name, Q5_5_quantity, Q5_5_price))
```

```
The total expense is 170.
The quantity for Dinner is missing and filled with 0.
The quantity for Concert is missing and filled with 1.
The price for Dinner is missing and filled with 1.
The price for Concert is missing and filled with 10.
```