AY: 2023-2024
MIDTERM | ECUEO412
Apr. 2024

L2-S4: Dept. of Electrical Engineering
Teacher: A. Mhamdi
Time Limit: 1h

This document contains **6** pages numbered from **1/6** to **6/6**. As soon as it is handed over to you, make sure it is complete. The **2** tasks are independent and can be treated in the order that suits you.

The following rules apply:

❶ **No document** is allowed in the examination room.

❷ **Any electronic material**, except basic calculator, is prohibited.

❸ **Mysterious or unsupported answers** will not receive full credit.

❹ **Round results** to the nearest thousandth *(i.e., third digit after the decimal point)*.

❺ **Task N°2:** Each correct answer will grant a mark with no negative scoring.

Task N°1

⌛ 25mn | (6 points)

Let's say we have a web application called app. The contents of both `*.jl` and `*.jl.html` files are given hereafter.

```
#= app.jl =#
using GenieFramework
@genietools

@app begin
    @in N::Int32 = 1000
    @in amp::Float32 = 0.25
    @in freq::Int32 = 1
    @out my_sine = PlotData()

    @onchange N, amp, freq begin
        x = range(0, 1, length=N)
        y = amp*sin.(2* *freq*x)
        my_sine = PlotData(x=x, y=y,
            plot=StipplePlotly.Charts.PLOT_TYPE_LINE)
    end
end
@page("/", "app.jl.html")
```

```
<!-- app.jl.html -->
<header class="st-header q-pa-sm">
   <h1 class="st-header__title text-h3" Sinewave Dashboard </h1>
</header>

<div class="row">
   <div class="st-col col-12 col-sm st-module">
      <p><b># Samples</b></p>
      <q-slider v-model="N"
         :min="10" :max="1000"
         :step="10" :label="true">
      </q-slider>
   </div>
   <div class="st-col col-12 col-sm st-module">
      <p><b>Amplitude</b></p>
      <q-slider v-model="amp"
         :min="0" :max="3"
         :step=".5" :label="true">
   </q-slider>
   </div>
   <div class="st-col col-12 col-sm st-module">
      <p><b>Frequency</b></p>
      <q-slider v-model="freq"
         :min="0" :max="10"
         :step="1" :label="true">
   </q-slider>
   </div>
</div>

<div class="row">
   <div class="st-col col-12 col-sm st-module">
      <p><b>Sinewave</b></p>
      <plotly :data="my_sine"> </plotly>
   </div>
</div>
```


(a) (3 points) Add a `phase` input to `app.jl` file. Its type and default value are `Float32` and $\frac{\pi}{4}$ respectively.
(You are not required to re-write the entire code. Document any modifications or additions you make, explaining your changes.)

```
@in phase::Float32 = π/4
@onchange N, amp, freq, phase begin
    y = amp*sin.(2*π*freq*x .+ phase)
```

(b) (3 points) The input `phase` is a slider that ranges between $-\pi$ and π, by a step size of $\pi/100$. Update the `html` file accordingly.

```
<div class="st-col col-12 col-sm st-module">
    <p><b>Phase</b></p>
    <q-slider v-model="phase"
        :min="-3.14" :max="3.14"
        :step="0.03" :label="true">
    </q-slider>
</div>
```

AY: 2023-2024	Full Name: .....
L2-S4: Dept. of Electrical Engineering	ID: .....
Midterm | ECUEO412	Class: .....
Apr. 2024	Room: .....
Teacher: A. Mhamdi	Time Limit: 1h

✂----------

Answer Sheet

Task N°2 ⌛ 35mn | (14 points)

(a) (1 point) What is `Genie`?

- ◯ A package for data visualization in `Julia`
- ◯ A machine learning library in `Julia`
- √ A web development framework in `Julia`
- ◯ A package for scientific computing in `Julia`

(b) (1 point) Which of the following features does `Genie` provide for web development?

- ◯ Routing and request handling
- ◯ Database integration
- ◯ Templating engine
- √ All of the above

(c) (1 point) How can you install `Genie` in `Julia`?

- √ `using Pkg; pkg"add Genie"`
- √ `import Pkg; Pkg.add("Genie")`
- □ `using Pkg; Pkg.install("Genie")`
- □ `pkg.add("Genie")`
- □ `pkg.install("Genie")`

(d) (1 point) What is the output of the following code?

```
1 str = "Julia"
2 print(str[2:4])
```

◯ `ul` ◯ `lia` √ `uli` ◯ `Julia`

(e) (1 point) What is the result of the following code?

DO NOT WRITE ANYTHING HERE

✂----------

```
x = 2 + 3im
y = 4 - 2im
z = x * y
```

◯ 10 + 2im ◯ 8 + 10im √ 14 + 8im ◯ 14 - 8im

(f) (1 point) What is the index number of the first element in Julia.

√ begin □ -1 □ 0 √ 1

(g) (1 point) What is the output of the code below?

```
x = 3 + 4im
print(real(x))
```

√ 3 ◯ 4 ◯ 7 ◯ 4im

(h) (1 point) What is the output of the code below?

```
x = 3 + 4im
print(imag(x))
```

◯ 3 √ 4 ◯ 7 ◯ 4im

(i) (1 point) What is the result of the following expression?

```
sqrt(-1+0im)
```

◯ 1 ◯ -1 √ 0 + 1im ◯ undefined

(j) (1 point) What is the output of the code below?

```
x = 5//7
y = 15//21
print(x == y)
```

◯ error ◯ undefined √ true ◯ false

(k) (1 point) The value of `result` is "62".

```
add(x, y=3) = x+y
square(x) = x^2
subtract(x, y=2) = x-y

result=5 |> add |> square |> subtract
print(result)
```

DO NOT WRITE ANYTHING HERE

✂----------

(l) (1 point) What is the main advantage of multiple dispatch in `Julia`.

- ○ It reduces code size
- ○ It makes functions run faster
- √ It allows to have multiple implementations based on argument types
- ○ Checks for type errors

(m) (2 points) What will be the output of the `greet` function after each call.

```julia
function greet(name::String)
    println("Hello, $name")
end

function greet(names::Vector{String})
    for name in names
        greet(name)
    end
end

function greet(name::Symbol)
    println("Hey there, $name")
end

greet("Ahmed")
greet(["Tracy", "Sara"])
greet(:student)
```

When we call the `greet` function with different argument types, `Julia` automatically dispatches to the appropriate method based on the types of the arguments.

```
Hello, Ahmed
Hello, Tracy
Hello, Sara
Hey there, student
```