PROBABILIDAD Y ESTADÍSTICA 1
Laboratorio de programación: github y shiny

Elementos básicos de programación desde la línea de comando

Crear un usuario gratuito en Posit y en github ejecutar en consola los siguientes comandos **desde la línea de comando**

```
# 1. Crear un archivo desde la terminal de posit dentro de un repositrio de git

mkdir app
cd app/
git init
git config --global user.email "su.correo@universidad.edu.co"
touch app.R
vim app.R
# Escribir su nombre como comentario (# Nicolas).
# --- 1. Para poder escribir, debe presionar la letra i
# --- 2. Una vez escrito. Presionar esc
# --- 3. Escribir :wq!
# --- 4. Verificar el resultado
cat app.R

# 2. Versionar el archivo usando git
git status
git add app.R
git status
git branch -M main
git commit -m 'Primer commit'

# 3. Crear un repositorio público usando github (sin readme.md).
#    En este caso es urosario_app.git

git remote add origin https://github.com/uranlopezl/urosario_app.git

# 4. Enviar los resultados 'locales' al repo remoto
# --- 1. Desde github, vaya a settings (globales, no las del repo creado)
# --- 2. En el panel izquierdo, baje hasta developer settings (simbolizado mediante
'<>')
# --- 3. Vaya a Personal access tokens. Tokens (classic)
# --- 4. Genere un nuevo token. Dele todos los permisos
# ------- (esto en general NO es recomendado) y guarde el token generado.

git push --set-upstream origin master
# Al hacer el push, le va a pedir user y psw.
# Use el user creado y como psw el token generado
```

La siguiente sección se debe ejecutar **desde Rstudio**. Abriendo el archivo `app.R`. Puede usar la interfaz gráfica de posit. No hace falta que lo haga desde cmd.

```
# Nicolas
# install.packages("shiny")
library(shiny)
library(ggplot2)

# It defines the user interface, the HTML webpage that humans interact with.
ui <- fluidPage(
  tags$h3("Scatter plot generator"),
  selectInput(inputId = "x", label = "X Axis", choices = names(mtcars), selected = "mpg"),
  selectInput(inputId = "y", label = "Y Axis", choices = names(mtcars), selected = "hp"),
  plotOutput(outputId = "scatterPlot")
)

# It specifies the behavior of our app by defining a server function.
server <- function(input, output){
  data <- reactive({mtcars})
  output$scatterPlot <- renderPlot({
    ggplot(data = data(), aes_string(x = input$x, y = input$y)) +
      geom_point() +
      theme_classic()
  })
}

# Construct and start a Shiny application from UI and server.
shinyApp(ui = ui, server = server)
```

Ejercicio

Vuelva ahora a la línea de comando y envíe a la ubicación remota el archivo `app.R` resultante.