Teor´ıa de la computación
Noviembre , 2025

Laboratorio 12

Para este laboratorio cree un repositorio de github

- Grabe un video de no m´a s de 10 minutos donde muestre la ejecución de su programa para el ejercicio1. Súb al o a YouTube como video no listado y adjúntelo en el README. de su repositorio.
- Para este ejercicio utilice cualquier otro lenguaje que permita programación funcional que no sea python.
- Para los incisos que no involucren la generación de códi go, cree una carpeta por separado en su repositorio y coloque las respuestas en un documento con formato PDF

Link de github: https://github.com/can23384/Laboratorio_12_TC

Link de video de ejecución: https://youtu.be/AKxpNa-Q2PY

Problema 1: 25%

- Escriba la reducci´on-β de la operaci´on l´ogica NOT
  NOT TRUE
  = (λb. b FALSE TRUE) TRUE
  →β TRUE FALSE TRUE
  = (λt.λf. t) FALSE TRUE
  →β (λf. FALSE) TRUE
  →β FALSE

  NOT FALSE
  = (λb. b FALSE TRUE) FALSE
  →β FALSE FALSE TRUE
  = (λt.λf. f) FALSE TRUE
  →β (λf. f) TRUE
  →β TRUE

- Escriba y explique como se ver´ıa la recursi´on y los ciclos.
  La recursión consiste en que una función se llama a sí misma para procesar una estructura de datos o repetir un cálculo, reduciendo el problema hasta llegar a un caso base. Para permitir la recursión en el cálculo-λ, que no admite funciones nombradas, se emplea el combinador de punto fijo, que permite definir funciones que se auto-referencian. Además de la recursión directa, también se simulan “ciclos” mediante funciones de orden superior como map, filter o fold, las cuales aplican funciones a colecciones de valores paso a paso.

- Explique cuando es prudente usar este tipo de programaci´on y cuando no. De unejemplo para cada caso.
  La programación funcional es recomendable cuando se necesita trabajar con operaciones sobre datos sin modificar estados, ya que facilita la claridad, la paralelización y las pruebas. Es ideal para tareas como análisis y transformación de información donde el flujo es predecible y basado en funciones puras.
  Ejemplo: procesamiento de grandes volúmenes de datos en Spark o análisis de listas con map y filter.

  Por otro lado, no es la mejor opción cuando se requiere manipulación constante del estado o rendimiento muy alto sobre estructuras mutables, como en programas interactivos o de

tiempo real. En estos casos, el estilo imperativo suele ser más natural y eficiente.
Ejemplo: un motor de videojuegos que actualiza posiciones, colisiones y físicas en cada frame.

Teor´ıa de la computación
Noviembre , 2025

Problema 2: 25%

Escribir un programa que ordene una lista de diccionarios con respecto a un key indicado, usando funciones lambda.

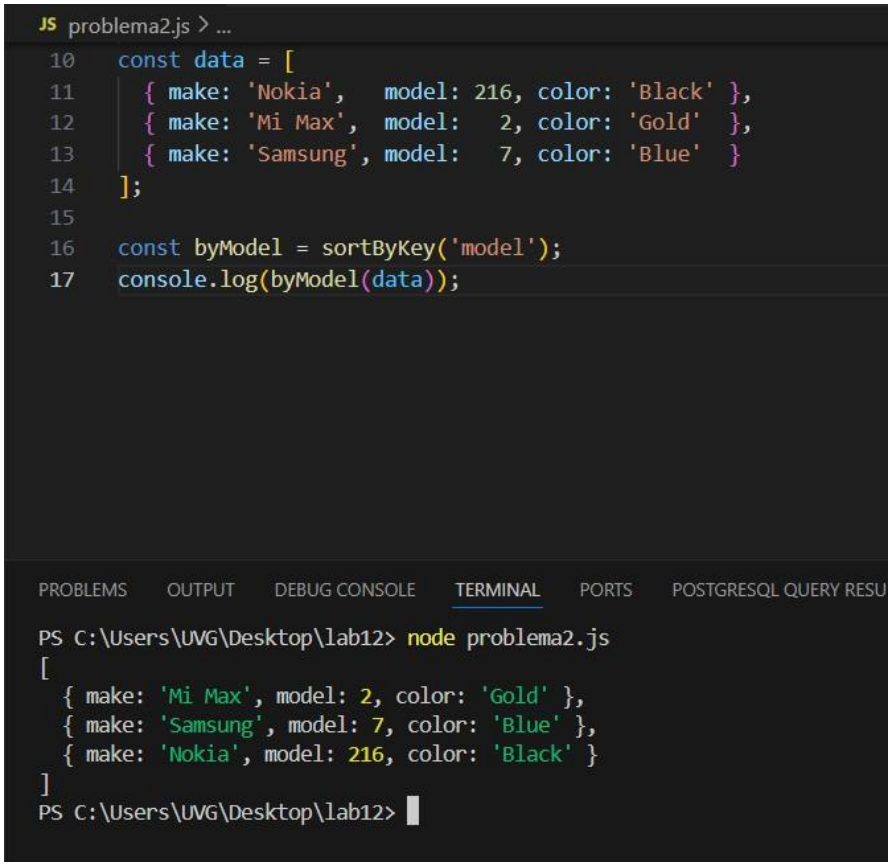
Ejemplo

Si la lista original de diccionarios es:

```
[
    {’make’: ’Nokia’, ’model’: 216, ’color’: ’Black’},
    {’make’: ’Mi Max’, ’model’: 2, ’color’: ’Gold’},
    {’make’: ’Samsung’, ’model’: 7, ’color’: ’Blue’}
]
```

La salida (si ordenamos con respecto a key = ’model’) debe ser:

```
[
    {’make’: ’Mi Max’, ’model’: 2, ’color’: ’Gold’},
    {’make’: ’Samsung’, ’model’: 7, ’color’: ’Blue’},
    {’make’: ’Nokia’, ’model’: 216, ’color’: ’Black’}
]
```



Teor´ıa de la computación
Noviembre , 2025

Problema 3: 25%

Se tiene una matriz X (definida como lista de listas). Escribir un programa que calcule la matriz transpuesta X^T, usando funciones lambda. La salida debe darse también como una lista de listas.

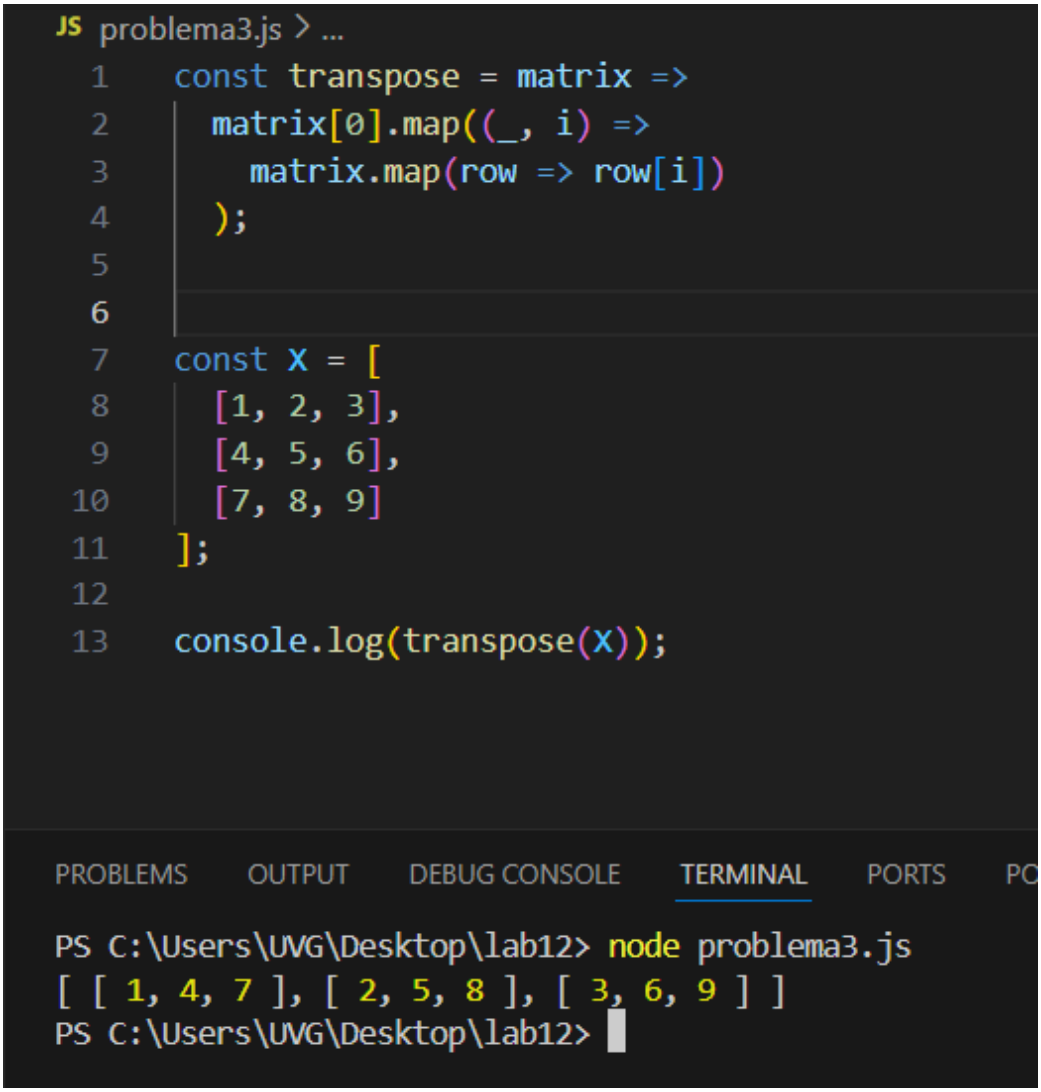
Ejemplo

Si la matriz original es:

$$X = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

Entonces la salida es:

$$X^T = \begin{bmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{bmatrix}$$



Teor´ıa de la computación
Noviembre , 2025

Problema 4: 25%

Escribir un programa que elimine elementos indicados de una lista, usando funciones lambda.
Los elementos a borrar deben indicarse como una lista, y la salida debe ser la lista de elementos remanentes.

Ejemplo

Si la lista inicial es:

```
1 ['rojo', 'verde', 'azul', 'amarillo', 'gris', 'blanco', 'negro']
```

Y la lista de elementos a borrar es:

```
1 ['amarillo', 'cafe', 'blanco']
```

Entonces la salida debe ser:

```
1 ['rojo', 'verde', 'azul', 'gris', 'negro']
```

Nota: Observe que 'café' no estaba en la lista original, por lo que simplemente se ignora.