Los primeros registros

Una historiadora está recopilando información acerca de distintos monumentos a lo largo y ancho del mundo . En principio solo quiso saber el nombre, ubicación, y año de construcción de cada monumento.

Para eso almacenó cada dato en una variable:

```
nombreEstatuaDeLaLibertad = "Estatua de la Libertad";
locacionEstatuaDeLaLibertad = "Nueva York";
anioDeConstruccionEstatuaDeLaLibertad = "1886";
nombreCristoRedentor = "Cristo Redentor";
locacionCristoRedentor = "Rio De Janeiro";
anioDeConstruccionCristoRedentor = "1931";
```

Ahí es cuando se dio cuenta que no era conveniente : si bien la información entre las variables estaba relacionada, la estaba almacenando por separado. Entonces pensó: ¿no existirá alguna forma de representar las distintas características o propiedades de una misma cosa de forma agrupada?

> Luego de investigar un poco, encontró una mejor manera para guardar la información de los monumentos. Probá en la consola escribiendo:
>
> ```
> estatuaDeLaLibertad
> cristoRedentor
> torreEiffel
> tajMahal
> coliseo
> ```

Consola Biblioteca

```
>
```

Esta guía fue desarrollada por Felipe Calvo bajo los términos de la Licencia Creative Commons Compartir-Igual, 4.0.

© 2015-2022 Ikumi SRL

Información importante

Términos y Condiciones

Reglas del Espacio de Consultas

Tu propio monumento

Los monumentos que probaste en el ejercicio anterior están representados como *registros*, y cada una de sus características (nombre, locación, año de construcción) son *campos* del registro. Por cierto, ¡podemos crear registros de cualquier cosa, con los campos que querramos!

Por ejemplo, podríamos almacenar un libro de modo que cada campo del registro fuese alguna característica: su título, su autor, su fecha de publicación, y más.

> ¡Es tu momento del monumento! Guardá en las variables `torreAzadi` y `monumentoNacionalALaBandera` registros de esos monumentos, oriundos de las ciudades de `Teherán, Irán` y `Rosario, Argentina` respectivamente. ¿Te animás a investigar en qué año se terminaron de construir para completar ese campo?

¡Dame una pista!

Solución | Biblioteca | Consola

```
let torreAzadi = { nombre: "Torre Azadi", locacion: "Teherán, Irán", anioDeConstruccion: 1971 };

let monumentoNacionalALaBandera = { nombre: "Monumento Nacional A La Bandera", locacion: "Rosario, Argentina", anioDeConstruccion: 1957 };
```

Enviar

¡Muy bien! Tu solución pasó todas las pruebas

¡Buenas habilidades de búsqueda! Los registros, al igual que las listas, son una *estructura de datos* porque nos permiten organizar información. Pero ¿en qué se diferencia un registro de una lista?

En las listas podemos guardar muchos elementos de un mismo tipo que representen una misma cosa (por ejemplo todos números, o todos strings). No existen límites para las listas: pueden tener muchos elementos, ¡o ninguno!

En un registro vamos a guardar información relacionada a una única cosa (por ejemplo **un** monumento o **una** persona), pero los tipos de los campos pueden cambiar. Por ejemplo, el nombre y la ubicación de un monumento son strings, pero su año de construcción es un número.

Esta guía fue desarrollada por Felipe Calvo bajo los términos de la Licencia Creative Commons Compartir-Igual, 4.0.

© 2015-2022 Ikumi SRL

Información importante

Términos y Condiciones

Reglas del Espacio de Consultas

Accediendo al campo

Cuando consultaste los registros existentes, se veía algo parecido a lo siguiente:

3. Accediendo al campo

```
> tajMahal
{ nombre: "Taj Mahal", locacion: "Agra, India", anioDeConstruccion: 1653 }
```

Esa consulta era porque estábamos viendo al registro `tajMahal` completo, incluyendo todos sus campos. ¡Pero también se puede consultar por un campo particular! Mirá :

```
> tajMahal.locacion
"Agra, India"
> tajMahal.anioDeConstruccion
1653
```

> Declaramos los planetas `mercurio`, `marte` y `saturno` como registros con la siguiente información: `nombre`, `temperaturaPromedio` y si `tieneAnillos`. ¡Probalos en la consola!

Consola | Biblioteca

```
>
```

Esta guía fue desarrollada por Felipe Calvo bajo los términos de la Licencia Creative Commons Compartir-Igual 4.0

Igual, 4.0.

© 2015-2022 Ikumi SRL

Información importante

Términos y Condiciones

Reglas del Espacio de Consultas

Temperatura de planeta

Ahora que agregamos registros de planetas, ¡trabajemos un poco con ellos!

4. Temperatura de planeta

Definí una función `temperaturaDePlaneta` que reciba como argumento un registro de planeta y retorne un string que indique su nombre y su temperatura promedio. ¡Tiene que funcionar para cualquier planeta! Por ejemplo:

```
> temperaturaDePlaneta(mercurio)
"Mercurio tiene una temperatura promedio de 67 grados"
> temperaturaDePlaneta(saturno)
"Saturno tiene una temperatura promedio de -139 grados"
> temperaturaDePlaneta(venus)
"Venus tiene una temperatura promedio de 462 grados"
```

¡Dame una pista!

Solución | Biblioteca | Consola

```
1 function temperaturaDePlaneta(planet){
2  return (planet.nombre+" tiene una temperatura promedio de "+
  planet.temperaturaPromedio  +" grados");
3 }
```

Enviar

¡Muy bien! Tu solución pasó todas las pruebas

Esta guía fue desarrollada por Felipe Calvo bajo los términos de la Licencia Creative Commons Compartir-Igual, 4.0.

© 2015-2022 Ikumi SRL

Información importante

Términos y Condiciones

Reglas del Espacio de Consultas

Moviendo archivos

Por el momento estuvimos creando y consultando registros. ¿No sería interesante poder... modificarlos?

La sintaxis para modificar campos de registros es muy similar a lo que hacemos para cambiar los valores de las variables. Por ejemplo, para cambiar la temperatura de un planeta:

```
saturno.temperaturaPromedio = -140;
```

Ahora imaginá que tenemos un registro para representar un archivo, del que sabemos su ruta (dónde está guardado) y su fecha de creación. Si queremos cambiar su ruta podemos hacer...

```
> leeme
{ ruta: "C:\leeme.txt", creacion: "23/09/2004" }

> moverArchivo(leeme, "C:\documentos\leeme.txt")
```

Luego el registro `leeme` tendrá modificada su ruta:

```
> leeme
{ ruta: "C:\documentos\leeme.txt", creacion: "23/09/2004" }
```

> ¡Es tu turno! Definí el procedimiento `moverArchivo`, que recibe un registro y una nueva ruta y modifica el archivo con la nueva ruta.

Solución | Biblioteca | Consola

```
1 function moverArchivo(registro, nueva_ruta){
2   registro.ruta= nueva_ruta;
3 }
4
```

Enviar

¡Muy bien! Tu solución pasó todas las pruebas

Esta guía fue desarrollada por Felipe Calvo bajo los términos de la Licencia Creative Commons Compartir-Igual, 4.0.

© 2015-2022 Ikumi SRL

Información importante

Términos y Condiciones

Reglas del Espacio de Consultas

Registros de dos milenios

En el ejercicio anterior modificamos la ruta del registro, pero no utilizamos su fecha de creación. ¡Usémosla! Queremos saber si un archivo es del milenio pasado, lo que ocurre cuando su año es anterior a 2000.

Definí la función `esDelMilenioPasado`, que recibe un archivo y retorna un booleano.

```
> esDelMilenioPasado({ ruta: "D:\fotonacimiento.jpg", creacion: "14/09/1989" })
true
```

¡Dame una pista!

Solución | Biblioteca | Consola

```
1 function esDelMilenioPasado(archivo){
2   return 2000>anio(archivo.creacion);
3 }
```

Enviar

¡Muy bien! Tu solución pasó todas las pruebas

Esta guía fue desarrollada por Felipe Calvo bajo los términos de la Licencia Creative Commons Compartir-Igual, 4.0.

© 2015-2022 Ikumi SRL

Información importante

Términos y Condiciones

Reglas del Espacio de Consultas

Postres complejos

Unos ejercicios atrás te contamos la diferencia entre listas y registros. ¡Pero eso no significa que no podamos usar ambas estructuras a la vez!

Por ejemplo, una lista puede ser el campo de un registro. Mirá estos registros de postres, de los cuales sabemos cuántos minutos de cocción requieren y sus ingredientes:

```
let flanCasero = { ingredientes: ["huevos", "leche", "azúcar", "vainilla"], tiempoDeCoccion: 50 }
let cheesecake = { ingredientes: ["frambuesas", "queso crema"], tiempoDeCoccion: 80 }
let lemonPie = { ingredientes: ["jugo de limón", "almidón de maíz", "leche", "huevos"], tiempoDeCoccion: 65 }
```

Definí la función `masDificilDeCocinar`, que recibe dos registros de postres como argumentos y retorna el que tiene más ingredientes de los dos.

```
> masDificilDeCocinar(flanCasero, cheesecake)
{ ingredientes: ["huevos", "leche", "azúcar", "vainilla"], tiempoDeCoccion: 50 }
```

¡Dame una pista!

Solución | Biblioteca | Consola

```
function masDificilDeCocinar(postre1, postre2){
    if (longitud(postre1.ingredientes)>longitud(postre2.ingredientes)) {
    return postre1;
    } else {
    return postre2;
    }
}
```

Enviar

¡Muy bien! Tu solución pasó todas las pruebas

Esta guía fue desarrollada por Felipe Calvo bajo los términos de la Licencia Creative Commons Compartir-Igual, 4.0.

© 2015-2022 Ikumi SRL

Información importante

Términos y Condiciones

Reglas del Espacio de Consultas

Listas de registros

En el ejercicio anterior te mostramos que un registro puede tener una lista entre sus campos. ¿Y al revés? ¿Podemos tener una lista de registros?

¡Sí! Así como trabajamos con listas de números, booleanos, strings o más listas, también podemos listar registros. Se puede hacer todo lo que hacías antes, como por ejemplo `remover`, saber su `longitud` o preguntar por el elemento de cierta posición utilizando los corchetes `[]`.

> Probá en la consola las listas `postresFavoritos` y `monumentosDeAmerica`. Hay un postre que no mostramos antes, ¿te das cuenta cuál es solamente leyendo sus ingredientes?

Consola | Biblioteca

```
>
```

Esta guía fue desarrollada por Felipe Calvo bajo los términos de la Licencia Creative Commons Compartir-Igual, 4.0.

© 2015-2022 Ikumi SRL

Información importante

Términos y Condiciones

Reglas del Espacio de Consultas

60 dulces minutos

A veces no sólo queremos comer algo rico, sino que queremos comerlo lo antes posible.

> Definí el procedimiento `agregarAPostresRapidos` , que toma una lista con postres rápidos y un postre por parámetro. Si el tiempo de cocción es de una hora o menos, se agrega el registro a la lista.

¡Dame una pista!

Solución | Biblioteca | Consola

```
function agregarAPostresRapidos(lista_postres_rapidos,postre_parametro){
   if(postre_parametro.tiempoDeCoccion <= 60){
   agregar(lista_postres_rapidos,postre_parametro);
   }
}
```

Enviar

¡Muy bien! Tu solución pasó todas las pruebas

Esta guía fue desarrollada por Felipe Calvo bajo los términos de la Licencia Creative Commons Compartir-Igual, 4.0.

© 2015-2022 Ikumi SRL

Información importante

Términos y Condiciones

Reglas del Espacio de Consultas

Hay un registro en mi registro

¿Te acordás cuando vimos que una lista podía estar compuesta por otras listas? ¡Con los registros aplica la misma idea! Si tenemos alguna estructura de datos compleja, puede ocurrir que no alcance con representarla únicamente mediante strings, números, booleanos y listas, sino que necesitemos *otro* registro dentro.

¡No se puede vivir a base de postres! Bueno, quizás sí, pero mantengamos una alimentación saludable . Mediante un registro queremos modelar un menú completo: consiste en un plato principal , los vegetales de la ensalada que acompaña , y un postre como lo veníamos trabajando, es decir, sigue siendo un registro.

Por ejemplo, el siguiente es un menú con bife de lomo como plato principal, una ensalada de papa, zanahoria y arvejas como acompañamiento y un cheesecake de postre. Como el registro es un poco extenso, y para que sea más legible, lo vamos a escribir de la siguiente forma:

```
let menuDelDia = {
  platoPrincipal: "bife de lomo",
  ensalada: ["papa", "zanahoria", "arvejas"],
  postre: { ingredientes: ["queso crema", "frambuesas"], tiempoDeCoccion: 80 }
};
```

Averiguá qué devuelve el campo `ingredientes` del campo `postre` del registro `menuInfantil`. ¡Está un registro adentro del otro! La sintaxis es la siguiente:

```
menuInfantil.postre.ingredientes
```

Consola Biblioteca

```
>
```

Esta guía fue desarrollada por Felipe Calvo bajo los términos de la Licencia Creative Commons Compartir-Igual, 4.0.

© 2015-2022 Ikumi SRL

Información importante

Términos y Condiciones

Reglas del Espacio de Consultas

¡Azúcar!

Para terminar, trabajemos una vez más con los menúes.

> Definí un procedimiento `endulzarMenu` , que recibe un registro menú y le agrega `azúcar` a los ingredientes de su postre. Si ya tiene azúcar, no importa... ¡le agrega más!

¡Dame una pista!

Solución | Biblioteca | Consola

```
function endulzarMenu(registro_menu){
  agregar(registro_menu.postre.ingredientes,"azucar");
}
```

Enviar

¡Muy bien! Tu solución pasó todas las pruebas

Esta guía fue desarrollada por Felipe Calvo bajo los términos de la Licencia Creative Commons Compartir-Igual, 4.0.

© 2015-2022 Ikumi SRL

Información importante

Términos y Condiciones

Reglas del Espacio de Consultas