LA PROGRAMACIÓN: INICIATE EN UN MUNDO APASIONANTE

3. ESTRUCTURAS DE DATOS

Hasta ahora hemos visto cómo trabajar con 3 tipos de datos: las cadenas, los números y los booleanos. Con estos datos podemos hacer muchas cosas, pero sería interesante poder trabajar con colecciones de datos.

Ejemplos de situaciones que requieren el uso de colecciones de datos: cuando trabajamos con los meses del año, los contacto de la agenda del móvil, o, en nuestro ejemplo de la baraja, el listado de palos de la baraja.

Arrays

Los arrays o colecciones son estructuras de datos que agrupan elementos. Para declarar un array en JavaScript utilizamos los corchetes `[` y `]`, e indicaremos en el interior los elementos del array separados por coma `,`.

Ejemplo 1: arrays de números y de cadenas

```
var numeros = [1, 2, 3];
var palos = ["c", "d", "p", "t"];
```

Para acceder a una posición del array utilizamos corchetes `[` y `]` y dentro la posición a la que queremos acceder. Recordemos que las posiciones empiezan a contarse desde el 0. Ejemplo: `palos[0]` nos devolverá la cadena `"c"` correspondiente a corazones.

Para actualizar una posición del array, también podemos utilizar el operador `[]` para acceder a una posición y usar la asignación `=` para darle un nuevo valor. Ejemplo: `palos[0] = "nuevo palo";`

Ejemplo 2: añadir elementos a un array

```
var palos = [];
palos[0] = "c";
palos[1] = "d";
```

También podemos consultar el tamaño de un array utilizando la propiedad `length` (longitud en inglés) del mismo, a la que accedemos con el operador punto. Ejemplo: `palos.length` nos devuelve el valor 4.

Ejemplo 3: recorriendo un array

```
var cartas = [10, 3, 5, 7];
var total = 0;
for(var i=0; i<cartas.length; i = i + 1){
  total = total + cartas[i];
}
```

Aquí vemos cómo puede utilizarse la propiedad de longitud de un array para recorrerlo, usando un índice que se incrementa en uno en cada iteración. Este bucle nos permite sumar la puntuación total de nuestra mano de cartas, que almacenamos en la variable total.

Ejemplo 4: búsqueda en un array

```
var cartas = ["5d", "3t", "2c", "12c"];
var encontrado = false;
for(var i=0; i<cartas.length; i=i+1){
  if(carta[i] === "1c"){
     encontrado = true;
  }
}
```

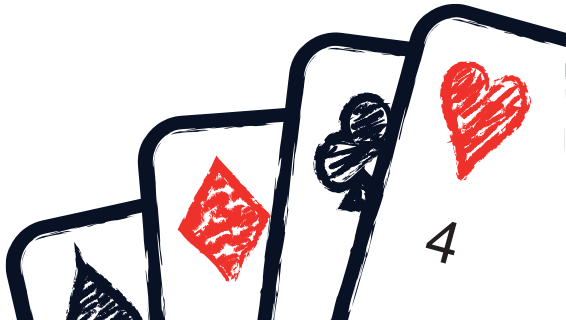
Usamos una variable booleana *encontrado* para almacenar el resultado de buscar una carta en un array de cartas. De esta forma, recorremos el array con un bucle y vamos comparando cada carta con la carta a buscar (1 de corazones). En caso de encontrarla, ponemos *encontrado* a true.

Ejemplo 5: inserción de valores en un array

```
var cartas = ["5d", "3t", "2c", "12c"];
var resultado = [];
for(var i=0; i<cartas.length; i=i+1){
  resultado[resultado.length] = cartas[i];
  if(cartas[i] === "3t"){
    resultado[resultado.length] = "1c";
  }
}
```

En este caso, es una buena práctica si vamos a modificar un array, crear una copia del mismo. Es decir, no vamos a modificar el array *cartas* sino que el resultado de la inserción lo vamos a crear en el array *resultado*.

Al comienzo el array resultado está vacío. Vamos recorriendo el array original y almacenando cada carta en la última posición, hasta que llegamos al 3 de tréboles (3t) en que añadiremos de forma adicional el as de corazones (1c). Al terminar, en la variable resultado tendremos el array modificado con una carta más en la posición posterior al 3 de tréboles.



Ejemplo 6: filtrado de un array

```
var cartas = ["5d", "3t", "2c", "12c"];
var resultado = [];
for(var i=0; i<cartas.length; i=i+1){
  if(cartas[i][1] === "d" || cartas[i][1] === "c"){
    resultado[resultado.length] = cartas[i];
  }
}
```

En este último ejemplo de arrays, iteramos una colección de cartas para detectar las que son de un palo de color rojo. Al igual que en el ejemplo anterior, vamos a crear un nuevo array *resultado* donde almacenaremos las cartas de color rojo que encontremos, es decir, las que sean de corazones o diamantes. Este ejemplo no termina de funcionar bien porque en el caso del rey de corazones, el palo no se encuentra en la posición 1 sino en la 2 (usamos 3 caracteres para definir cartas mayores que 9). Podemos modificar la condición para que dé cobertura a ambos casos, sólo para detectar cartas de diamantes: `if(cartas[i][1] === "d" || cartas[i][2] === "d")`. En el siguiente bloque veremos que para almacenar este tipo de datos, es mucho mejor usar un diccionario.

Diccionarios

Los diccionarios son estructuras que nos permiten almacenar datos estructurados. También se les llama *hash* en inglés. En los diccionarios la información se almacena con una clave y un valor. Por ejemplo, para almacenar una dirección postal tendríamos la información de un calle, una población, un código postal y un país.

Ejemplo 1: crear un diccionario

```
var carta = {palo: "c", valor: 1};
```

Usamos la notación de llaves `{ }` para declarar un diccionario, que consta de pares *clave: valor* separados por comas `,`. Para separar la clave de su valor usamos `:`. En JavaScript los diccionarios se llaman *Object* (objeto) y tienen más funcionalidades que veremos al final del curso.

Ejemplo 2: leer información de un diccionario

```
var carta = {palo: "c", valor: 1};
carta["palo"]
carta.palo
```

Tenemos dos maneras de acceder a los valores almacenados en un diccionario. En primer lugar, usando la misma notación que en los arrays con `[ ]` y poniendo entre comillas el nombre de la clave cuyo valor queremos obtener. La segunda forma es con la notación de punto, que separa el nombre del diccionario del de la clave cuyo valor queremos obtener. La notación más habitual es la notación con punto, ya que de esta forma distinguimos si trabajamos con un array o un diccionario.

Ejemplo 3: actualizar información de un diccionario

```
var carta = {palo: "c", valor: 1};
carta["palo"] = "d"
carta.palo = "t"
```

En este caso, usamos la misma notación anterior para modificar el valor del palo de una carta.

Ejemplo 4: array de diccionarios

```
var as = {palo: "c", valor: 1};
var rey = {palo: "t", valor: 12};
var cartas = [as, rey];
cartas[0].palo;
cartas[0].palo = "t";
```

En este ejemplo declaramos un array de cartas, siendo cada carta un diccionario que contiene un palo y un valor. En este ejemplo hemos declarado las cartas en variable separadas, pero podríamos declarar el array directamente con los diccionario sin las variables intermedias de as y rey. Luego accedemos al palo de la primera carta del array, y podemos modificarla.

Ejemplo 5: diccionario de arrays

```
var palos = {esp: ["c", "o", "e", "b"], poker: ["c",
"t", "p", "d"]}
palos.poker[3]
```

En este caso, hemos definido un diccionario de palos con el listado de palos de distintas barajas, la española y la de poker. Así, podemos acceder a cada listado, que es un array, y luego a cada posición dentro del array. Por ejemplo, accedemos a la última posición del listado de palos de la baraja de poker.