**CURSO DE ESTRUCTURAS DE DATOS CON JAVASCRIPT**

**¿QUÉ SON LAS ESTRUCTURAS DE DATOS?**

Una “estructura de datos” es una colección de valores, la relación que existe entre estos valores y las operaciones que podemos hacer sobre ellos; en pocas palabras se refiere a cómo los datos están organizados y cómo se pueden administrar. Una estructura de datos describe el formato en que los valores van a ser almacenados, cómo van a ser accedidos y modificados, pudiendo así existir una gran cantidad de estructuras de datos.

Si tenemos toda nuestra ropa desordena tirada por todas partes y hecho un montón. vamos a buscar la ropa que llevaremos para una fiesta nos tomara mucho tiempo y trabajo encontrar hasta el par de calcetines. Se pregunta ¿Cuál es la mejor manera de ordenar la ropa? Hay muchas formas y no hay una mejor que otra, esto solo depende del problema que se quiera resolver, para este caso basta con que se doble, cuelgue y organicé por tipo de ropa (camisa, pantalón, calcetines, etc.) hasta podría ordenar cada tipo por color, pero con lo anterior bastaría, este es solo un ejemplo, pero hay varios “modelos” con los que organizar nuestra ropa.

Con el ejemplo anterior podemos concluir que las estructuras de datos nos ahorran tiempo, espacio y son más eficientes a la hora de aplicar acciones sobre los datos.

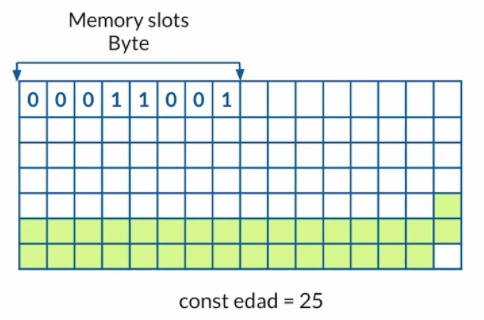
MEMORIA Y CÓMO SE GUARDAN LOS DATOS

Observando a la memoria desde un mayor nivel de abstracción, podemos imaginarla como si de un gran conjunto de celdas se tratara, cada una de estas celdas estaría marcada por un número que las identifica y es a este número al que podríamos referirnos como: dirección de memoria. Si tenemos la dirección de memoria, podemos ubicar fácilmente la celda en cuestión y acceder a lo que esta contiene.



Ya que la información de una computadora se mide por bytes, es lógico pensar que cada celda disponible en la memoria tiene el tamaño de un byte. No todos los valores que podemos guardar en memoria tienen el mismo tamaño, por tanto, estos pueden abarcar desde un simple byte hasta varios bytes (pueden tomar una o varias celdas).

Estas celdas se les suele llamar SLOTS de memoria, cada SLOT equivaldría a un BYTE(8 bits), la memoria puede representarse como una matriz de bits los cuales tienen una dirección asociada (addres).



Imagina por un momento que declaramos tres variables de tipo entero en nuestro programa y que cada entero tiene un tamaño de 4 bytes. Si suponemos que el sistema reserva memoria para estas variables en slots contiguos, tendríamos las siguientes direcciones que identifican al slot que ocupa cada variable:



Puedes observar que cada dirección de memoria está separada precisamente por 4 slots, si hubiéramos decidido que las variables fueran de algún tipo que tenga tamaño de un byte, cada dirección estaría separada por un solo slot.

Unidades de almacenamiento en la computadora:



ARRAYS

Los arrays son una estructura de datos lineal en la cual los elementos se guardan de forma ordenada y se identifican mediante un índice según su posición, como toda estructura de dato tiene asociada métodos que nos permite acceder o modificar sus valores.

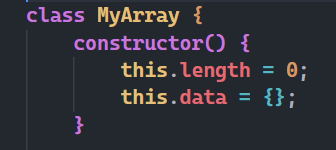
De manera general existen dos tipos de Arrays los estáticos y dinámicos, cada uno tiene sus ventajas y desventajas y su uso depende de la situación en la que se esté.

* *los arrays estáticos* desde el principio tiene definido su número de slot disponibles, y ya este valor no se puede modificar
* *los arrays dinámicos* trabajan siempre con duplicando la cantidad de slots que se le requiere, por ejemplo, si pides un arrays de 3 slops el mecanismo de arrays dinámicos va a proveer 6 slots para tener disponibles por si acaso tienda a crecer la cantidad de datos, por consiguiente, este tipo de arrays si se pueden modificar. (javascript usa arrays dinámicos)

CONSTRUYENDO NUESTRO ARRAY

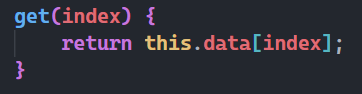
Lo que vamos hacer es construir nuestra propia implementación de arrays con sus respectivos métodos como el ***push, pop, delete****, etc.*

Primero declaramos la clase con el nombre ***MyArray,*** el constructor tendrá dos variables de clase, el primero nos ayudara a tener el conteo de cuantos elementos tenemos en el array como lo estamos inicializando su valor es de cero. Luego, declaramos la propiedad data donde se guardarán los elementos que se irán agregando.



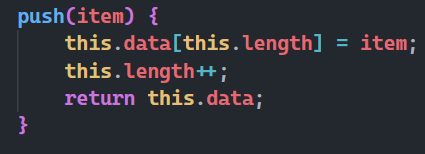
* método get().

Este método es uno de los más sencillos, el método recibe el argumento index como parámetro y retorna el valor de la posición index en data.



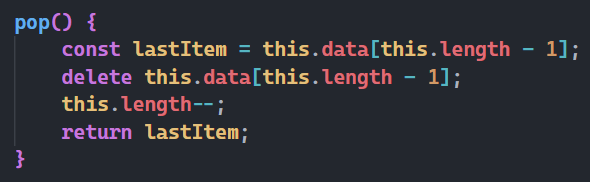
* método push()

Este método nos permite agregar nuevos elementos. Recibe el nuevo item como parámetro, este item se agrega a la variable en la posición ultima de data que equivale a length, luego aumentamos length y retornamos data al final.



* método pop()

El método pop elimina el ultimo elemento de un array. Antes de eliminar el item, lo guardamos en una variable. Con el operador delete eliminamos el elemento, como la longitud cambio se le resta el valor a length y por ultimo se retorna el valor que se eliminó.



* método delete()

El método delete como su nombre lo indica elimina un elemento dentro del array, recibe el index del elemento a eliminar. Primero se guarda el item que será retornado al final, luego se llama al método shiftIndex que nos permite reordenar el array desde la posición del item a eliminar y reduce length y por último dentro de delete se retorna el item eliminado.

