BFOA Y JAVASCRIP

BFOA JS IMPLEMENTATION

Erick Vazquez Hernandez

Resumen

Desarrollo del algoritmo BFOA, utilizando como lenguaje de programación Javascript. También como apoyo se utilizó HTML y Css , así como Bootstrap (framework) para el diseño de la página web.
Para la comprensión de algún tema, siempre se necesita, alguna forma de practica que pueda incrustarse en el alumno que este tomando este tema. Actualmente existen múltiples lenguajes que pueden soportar este tipo de algoritmos, pero se quedan solo en código, en un texto plano, que no representa la gran importancia de la investigación, o que no genera el interés del seguir buscando otro enfoque, distinto al tradicional.
Una forma de enfoque diferente es integrar el algoritmo a un sitio web, en el que el alumno, pueda desarrollar una pagina web , esto para que no se quede con la idea de solo un enfoque estático, si no, un enfoque, en el que se pueda abrir a cambios y desarrollo funcional.

Palabras Clave: JavaScrip, BFOA, Dinámica Ejemplificacion.

Abstract

Development of the BFOA algorithm, using Javascript as the programming language. HTML and CSS were also used as support, as well as Bootstrap (framework) for the design of the website.
In order to understand a topic, there is always a need for some form of practice that can be embedded in the student who is taking this topic. Currently, there are multiple languages that can support this type of algorithms, but they remain only in code, in a plain text, which does not represent the great importance of the research, or which does not generate the interest of continuing to look for another approach, different from the traditional one. A different form of approach is to integrate the algorithm into a website, in which the student can develop a web page, so that he does not stay with the idea of only a static approach, but rather, an approach in which he can be open to changes and functional development.

Keywords: JavaScript, BTOA, Dynamic Exemplification.

1. Introducción

Para el desarrollo se necesitara solo acceso a una computadora así como conexión a internet (Esto solo para Bootstrap)

2. Materiales y Métodos

-Computadora:
En esta se desarrollara el algoritmo y la pagina
-Navegador web:
Este puede ser cualquier navegador, ya que estos utilizan JavaScript como lenguaje natural.
-Entorno:
También se necesita crear una carpeta donde tengamos nuestras paginas juntas y los recursos para que pueda funcionar bien, y no existan errores ala hora de compilar el proyecto.
.GitHub:
Puede funcionar como base para que todos tengan acceso a lo mismo y puedan desarrollar mejoras.
Visual Studio Code:
VSC ofrece una manera efectiva de desarrollar código y poder ejecutarle de forma inmediata.

2.1 BFOA JS IMPLEMENTATION

Implementación del algoritmo fuera de los lenguajes comunes, estos no son malos ni mucho menos no eficientes, solo que representan un enfoque estático donde solo en texto plano se plasma el algoritmo.
Con una pagina web podemos acceder a un sinfín de información y estas tienen un sinfín de interacciones.
Al querer salir del enfoque tradicional, (Tradicional dentro de el entorno actual mío.) Buscar otra forma de hacer funcional y efectiva la comprensión de temas que no son tan amigables para un usuario común, o usuario curioso de información. Y la mejor manera es practica, donde se busca desarrollar una pagina web, sencilla, que no necesite demasiados recursos y que sea de fácil acceso.
Javascrip: como lenguaje nativo de los navegadores es una excelente opción ya que pudremos implenetar el código del algoritmo BFOA y integrarlo a una pagina web, donde podramos abrir la mente y no solo desarrollar para un usuario si no para múltiples usuarios curiosos que, talvez , sea su primera iteraccion con la información científica, o en este caso, un algoritmo científico que busca solucionar o atender problemáticas complejas.

2.3 función recopila los valores de los inputs que son las cadenas y los po ne en un array

```
function obtenerNombres() {
        var cantidadNombres = parseInt(document.getElementById("numeroCadenas").value);
  if (isNaN(cantidadNombres)) {
     console.error("El valor de 'numeroCadenas' no es un número válido.");
     return [];
  }
var nombres = [];
  for (var i = 0; i < cantidadNombres; i++) {
     var elemento = document.getElementById('Cadena' + i);
     if (elemento) {
        var nombre = elemento.value;
        nombre = nombre.toUpperCase();
        nombres.push(nombre.trim());
        console.log("Se encontro el elemento: " + elemento.value)
     } else {
        console.error("No se encontró el elemento con ID 'Cadena" + i + "'.");
     }
  }
  console.log(nombres);
  return nombres;
```

2.4 Balanceo

Esta función balancea las cadenas, recordemos quie se puede utilizar cuando se inicializa el código y cuando esta las iteraciones funcionando

```
function balanceo(array){
   // Encuentra la longitud máxima de todas las palabras en el array.
        const longitudMaxima = Math.max(...array.map((palabra) => palabra.length));

  // Rellena las palabras más cortas con guiones "-" al final hasta que tengan la longitud máxima.
```

```
 const palabrasIgualadas = array.map((palabra) =>
{
   while (palabra.length < longitudMaxima) {
    palabra += "-";
   }
   return palabra;
 });

 return palabrasIgualadas;
}
```

2.5 Función Posición Aleatoria

Esta función agraga los gabs (-) aleatoriamenta

```
function poscicionaleatoria(x){
   let poscicionAleatoria
     poscicionAleatoria=Math.floor(Math.random(
)*x)
   console.log(poscicionAleatoria)
   return poscicionAleatoria
  }
```

2.6 Función que compra las Palabras

```
function compararPalabras(array) {
   if (array.length < 2) {
    console.log("Se necesitan al menos dos
palabras para comparar.");
    return;
   }
   const primeraPalabra = array[0];
   const resultados = { iguales: 0, guiones: 0 };

   for (let i = 0; i < primeraPalabra.length; i++) {
    const caracterAComparar = primeraPalabra[i];

    if (caracterAComparar === '-') {
     resultados.guiones++;
    } else {
     for (let j = 1; j < array.length; j++) {
      if (caracterAComparar === array[j][i]) {
       resultados.iguales++;
      }
     }
    }
   }

   return resultados;
 }
```

2.6 Alta y baja permutación.

```
for (let i = 0; i < x; i++) {
        // Aplica la función agregar_gabs a
altaMutacion
     altaMutacion = altaMutacion.map((secuencia)
=> agregar_gabs(secuencia));

        // Aplica la función agregar_gabs a
bajaMutacion
    bajaMutacion = bajaMutacion.map((secuencia)
=> agregar_gabs(secuencia));
   }

    // Calcula el rendimiento de altaMutacion y
BajaMutacion
          const rendimientoAltaMutacion =
altaMutacion.map((secuencia) =>
compararPalabras(secuencia));
          const rendimientoBajaMutacion =
bajaMutacion.map((secuencia) =>
compararPalabras(secuencia));

    // Compara el rendimiento de altaMutacion y
BajaMutacion con el rendimiento actual
       const mejorRendimientoAltaMutacion =
Math.max(...rendimientoAltaMutacion);
       const mejorRendimientoBajaMutacion =
Math.max(...rendimientoBajaMutacion);

   // Compara con el rendimiento actual y devuelve
el mejor resultado
        if (mejorRendimientoAltaMutacion >
mejorRendimientoBajaMutacion &&
mejorRendimientoAltaMutacion >
mejorTemp.iguales) {
                                    return
altaMutacion[rendimientoAltaMutacion.indexOf(
mejorRendimientoAltaMutacion)];
    } else if (mejorRendimientoBajaMutacion >
mejorTemp.iguales) {
                                    return
bajaMutacion[rendimientoBajaMutacion.indexOf(
mejorRendimientoBajaMutacion)];
   } else {
    return Thebesth;
   }
```

}

2.6 Función Principal

Esta función engloba las demás y es la que manda a llamar el botón comparar del html.

Fuction BFOA(){

}

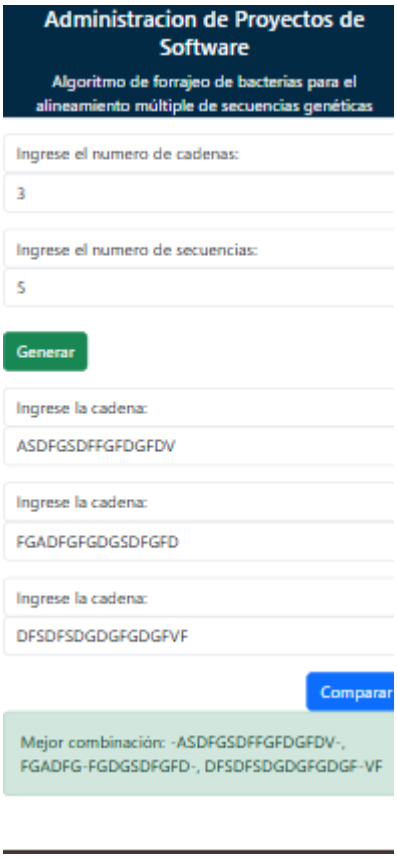

Figura 1. Resultado Pagina Web Responsiva . [1]

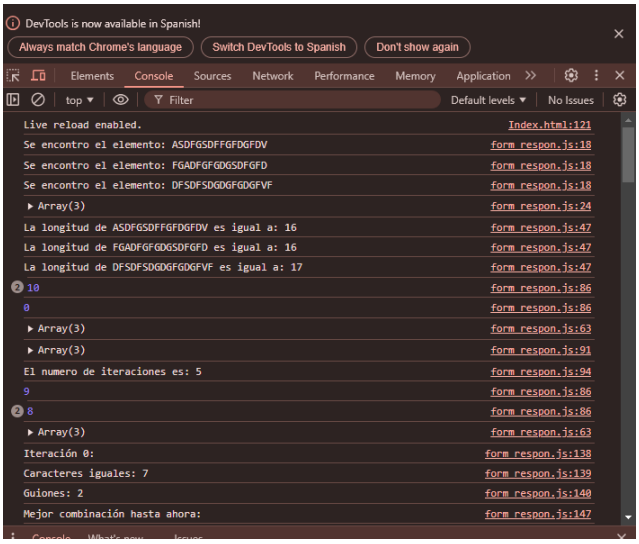

Figura 2. Resultado Pagina Web Consola . [1.2]

3. Resultados y Discusión

Como resultado principal es abrir un tema, y poder desglosarle he integrarlo a diferentes tecnologías actuales, para una mejor comprensión de estos.

4. Conclusiones

La información y el desarrollo no deben centrarse en solo una parte de la sociedad, si no en buscar acercar al toda la sociedad, en intereses de la búsqueda científica, y esto de manera amigable. Para no abrumar con tanta información innecesaria (Para un usuario o persona de la sociedad común) que solo pueda hacer perder el interés de esta.

Referencias

Github

[1] RigarSaltillo/AlgoritmoGenetico: Administracion de Proyectos de Software