

**Tabla de código:**

| **Paso** | **Descripción** |
| --- | --- |
| **1. Definir el alfabeto** | Definir una lista que contenga todas las letras del alfabeto en mayúsculas, incluyendo la "Ñ". |

abc = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'Ñ', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z']

**2. Inicializar las variables** | Asignar los elementos del DOM a las variables correspondientes.

textarea = document.querySelector("textarea");

inpText = document.querySelector("input[name='clave']");

form = document.querySelector("form");

resultado = document.querySelector("#resultado");

table = document.querySelector("#tablaMatrices");

**3. Función para validar entradas** | Crear una función que filtre los caracteres no válidos en las entradas y elimine espacios innecesarios.

function validarInput(input) {

input = input.replace(/[^A-Za-zÑ]/g, ''); // Reemplazar caracteres no válidos

return input.trim().replace(/\s+/g, ' '); // Eliminar espacios extra y recortar

}

**4. Al cargar el documento** | Al cargar la página, asignar los elementos a las variables y agregar eventos para la validación y manejo del formulario.

window.onload = function() {

// Asignar elementos a las variables

// Agregar eventos para validar y enviar el formulario

};

**5. Al enviar el formulario** | Cuando se envíe el formulario, prevenir el comportamiento por defecto y obtener los valores necesarios.

form.addEventListener("submit", function(e) {

e.preventDefault();

// Limpiar resultados y obtener clave, mensaje y opción seleccionada

});

**6. Validación de entradas** | Verificar si los campos de clave y mensaje están vacíos y si se ha seleccionado una opción, mostrando alertas en caso contrario.

if (!inpText.value || !textarea.value) {

alert("Por favor, ingrese tanto el mensaje como la clave.");

} else if (!document.querySelector("input[name='opcion']:checked")) {

alert("Seleccione una opción para continuar.");

}

**7. Crear una nueva clave** | Generar una nueva clave utilizando el mensaje y la clave, agregando los caracteres de la clave según corresponda.

// Lógica para combinar mensaje y clave

**8. Dependiendo de la opción seleccionada** | Si la opción es "1", cifrar el mensaje; si es "2", descifrar el mensaje.

if (opcionSeleccionada === "1") {

// Cifrar y mostrar mensaje cifrado

// Crear matriz de Vigenère

} else {

// Descifrar y mostrar mensaje descifrado

// Crear matriz de Vigenère para descifrar

}

**9. Función para cifrar** | Cifrar el mensaje utilizando la clave, iterando sobre los caracteres y calculando las posiciones en el alfabeto.

function cifrar(mensaje, clave) {

let resultado = "";

for (let i = 0; i < mensaje.length; i++) {

let mensajePos = abc.indexOf(mensaje[i]);

let clavePos = abc.indexOf(clave[i % clave.length]);

let cifrado = (mensajePos + clavePos) % abc.length;

resultado += abc[cifrado];

}

return resultado;

}

**10. Función para descifrar** | Similar a la función de cifrado, pero con una fórmula diferente para revertir el proceso.

function descifrar(mensaje, clave) {

let resultado = "";

for (let i = 0; i < mensaje.length; i++) {

let mensajePos = abc.indexOf(mensaje[i]);

let clavePos = abc.indexOf(clave[i % clave.length]);

let descifrado = (mensajePos - clavePos + abc.length) % abc.length;

resultado += abc[descifrado];

}

return resultado;

}

**11. Crear matriz de Vigenère** | Generar una matriz HTML que muestre cómo se cifra el mensaje, con cada fila representando un carácter de la clave.

function crearMatrizVigenere(mensaje, clave) {

// Crear una tabla HTML mostrando la matriz

}

**12. Crear matriz de Vigenère para descifrar** | Crear una matriz similar, pero para mostrar el proceso de descifrado.

function crearMatrizVigenereDescifrar(mensaje, clave) {

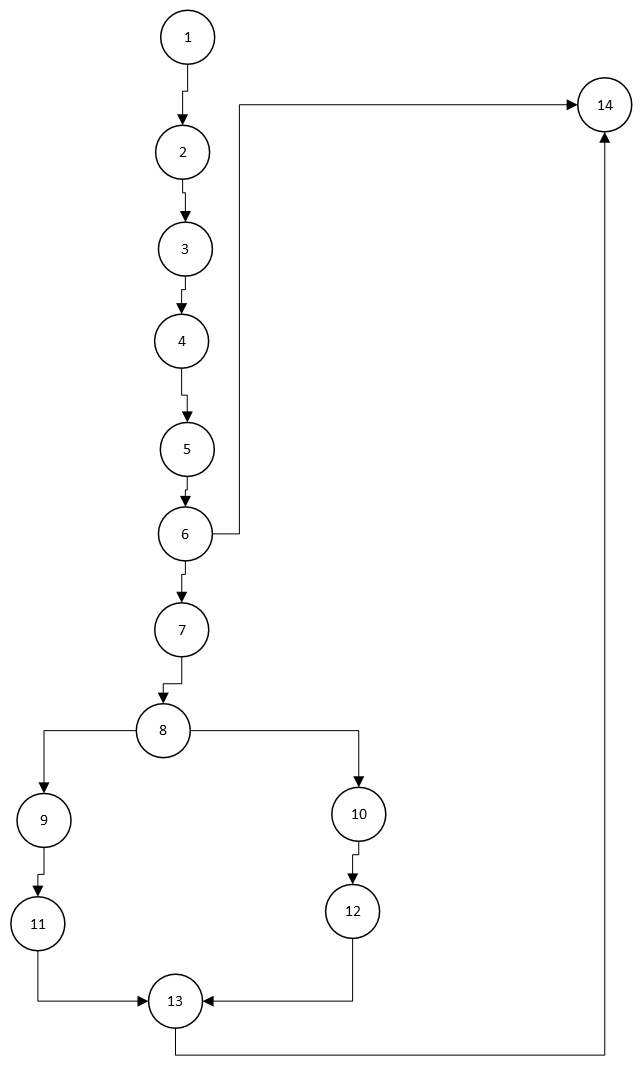
// Crear tabla similar a la de cifrado, pero para el proceso de descifrado

}

**13. Mostrar resultados** | Mostrar los resultados de las operaciones de cifrado o descifrado en la página.

resultado.textContent = "Mensaje: " + resultado;

Grafo de Flujo



* **Camino A (Cifrado exitoso):**
  + Ingreso de mensaje y clave → Validación exitosa → Opción 1 seleccionada → Cifrado → Mostrar resultado.
* **Camino B (Descifrado exitoso):**
  + Ingreso de mensaje y clave → Validación exitosa → Opción 2 seleccionada → Descifrado → Mostrar resultado.
* **Camino C (Validación fallida - mensaje vacío):**
  + Ingreso de mensaje vacío → Se muestra alerta de entrada vacía.
* **Camino D (Validación fallida - clave vacía):**
  + Ingreso de clave vacía → Se muestra alerta de entrada vacía.
* **Camino E (Validación fallida - opción no seleccionada):**
  + Ingreso de mensaje y clave → Validación exitosa → Opción no seleccionada → Se muestra alerta de opción no seleccionada.

**Total de nodos:**

* **Nodos (N):**
  1. Definir el alfabeto
  2. Inicializar variables
  3. Función para validar entradas
  4. Al cargar el documento
  5. Al enviar el formulario
  6. Validar entradas
  7. Crear nueva clave
  8. Opción seleccionada: Cifrar
  9. Opción seleccionada: Descifrar
  10. Función para cifrar
  11. Función para descifrar
  12. Crear matriz de Vigenère
  13. Crear matriz de Vigenère para descifrar
  14. Mostrar los resultados
  15. Fin del programa

**Total de nodos:** 15 (𝑁 = 15)

**Aristas (E):**

* Desde **Inicio** a **Definir el alfabeto**
* Desde **Definir el alfabeto** a **Inicializar variables**
* Desde **Inicializar variables** a **Función para validar entradas**
* Desde **Función para validar entradas** a **Al cargar el documento**
* Desde **Al cargar el documento** a **Al enviar el formulario**
* Desde **Al enviar el formulario** a **Validar entradas**
* Desde **Validar entradas** a **Fin del programa** (si se muestra alerta por vacío)
* Desde **Validar entradas** a **Crear nueva clave** (si es válido)
* Desde **Crear nueva clave** a **Opción seleccionada: Cifrar**
* Desde **Opción seleccionada: Cifrar** a **Fin del programa**
* Desde **Crear nueva clave** a **Opción seleccionada: Descifrar**
* Desde **Opción seleccionada: Descifrar** a **Fin del programa**
* Desde **Crear matriz de Vigenère** a **Fin del programa**
* Desde **Mostrar los resultados** a **Fin del programa**

**Total de aristas:** 18 (𝐸 = 18)

**Componentes conectados (P):**  
𝑃 = 1

**Cálculo de la complejidad:**

La fórmula para calcular el número de vértices 𝑉(G) es:

V(G)=E−N+2P=18−15+2(1)=18−15+2=5V(G) = E - N + 2P = 18 - 15 + 2(1) = 18 - 15 + 2 = 5V(G)=E−N+2P=18−15+2(1)=18−15+2=5

**Valor de V(G):** 5

Tablas de Pruebas

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Caso de Prueba | Entrada (Mensaje) | Entrada (Clave) | Opción Seleccionada | Salida Esperada |
| 1 | "MUNDO" | "CLAVE" | 1 | Mensaje cifrado: " ÑFNYS " |
| 2 | "HOLA MUNDO" | "PYTHON" | 1 | Mensaje cifrado: "WNEH AHCBI " |
| 3 | "Julian es gei" | "enefecto" | 1 | Mensaje cifrado: "NHONEO XH KQM" |
| 4 | "HOLA" | "LONGKEY" | 1 | Mensaje cifrado: "**RDXG**" |
| 5 | "HOLA" | "K" | 2 | Mensaje descifrado: "XFBQ" |
| 6 | "cffff” | "k" | 1 | Mensaje cifrado: "MOOOO” |
| 7 | "PRUEBA" | "CONTRASEÑA" | 2 | Mensaje descifrado: "ÑDILKA" |

Ejemplos pruebas con mocha

Texto

Descripción generada automáticamente Texto

Descripción generada automáticamente