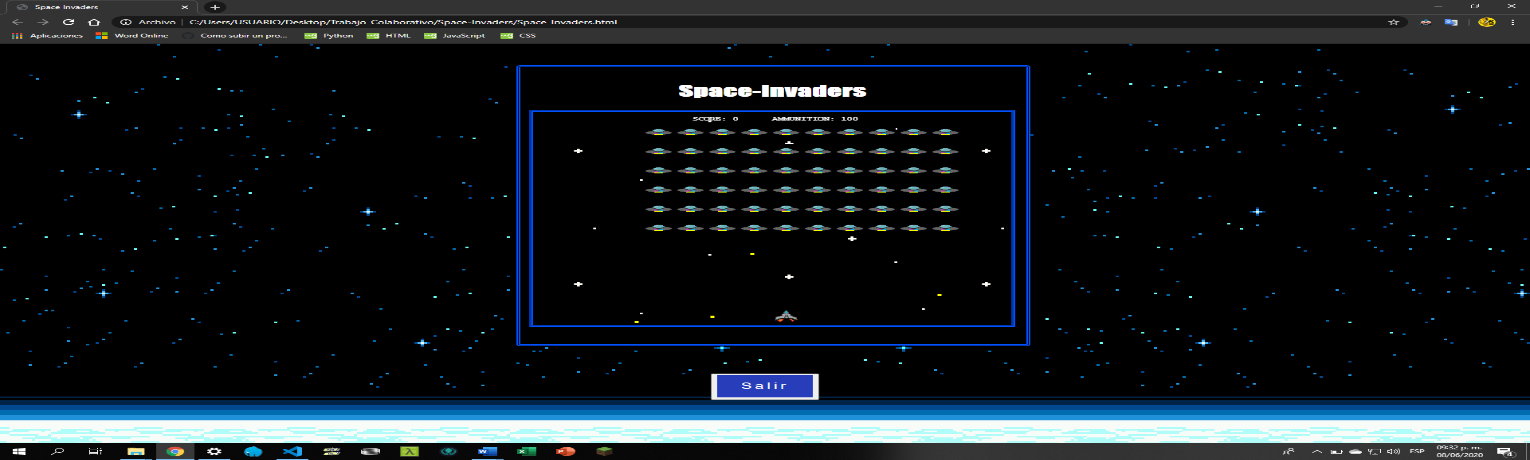
**Documentación Externa.**

**Documento Escrito.**

* **Descripción paso a paso.**

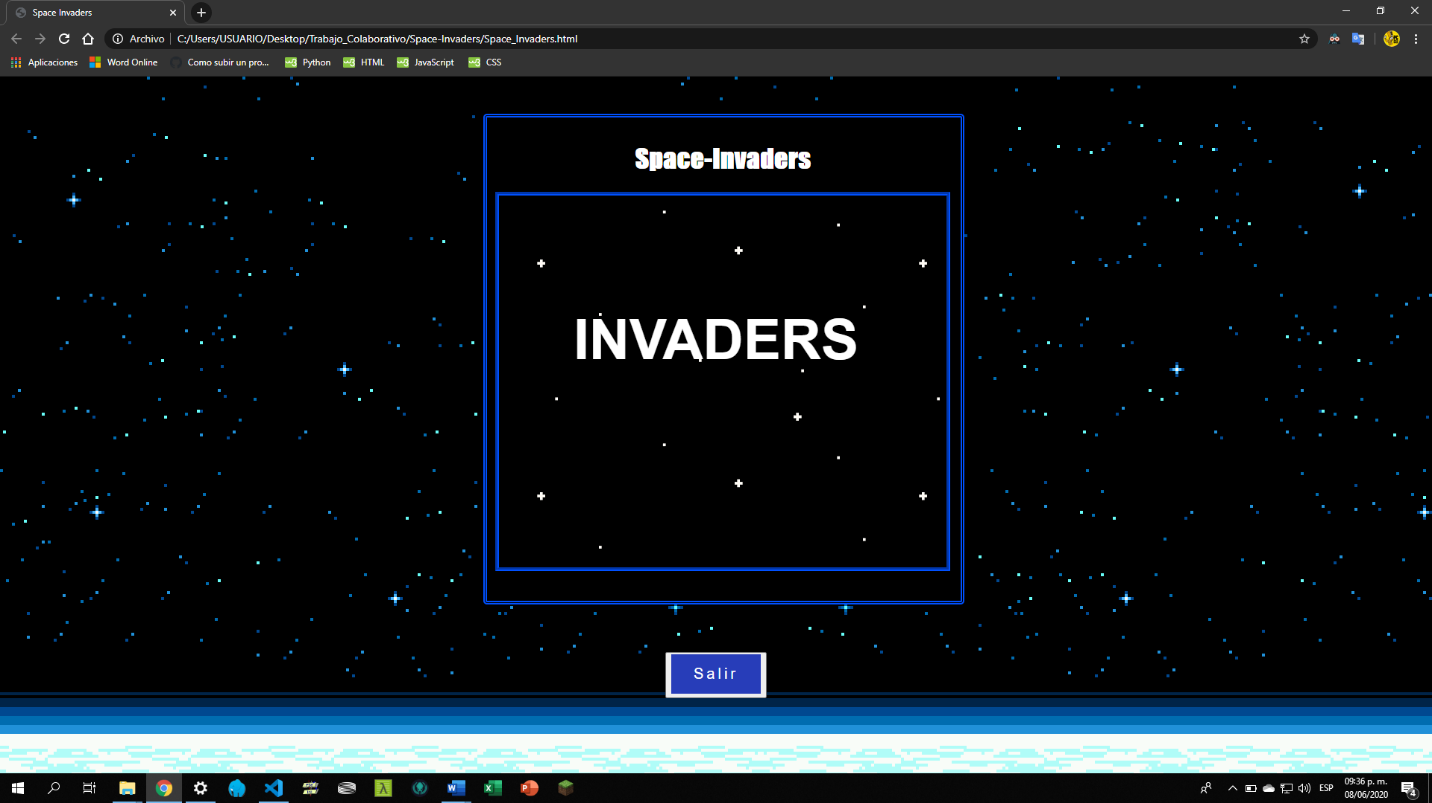
Este software requiere de la lectura de un archivo “js” y un “css” para poder funcionar correctamente, así mismo hace uso de una tabla donde se muestra el “Canvas” (que se utilizara para el dibujado de nuestro juego en el “body” del “html”), así mismo un botón que nos permita salir a la pagina principal. Para el archivo js tenemos una función para cargar la pantalla en nuestro html, así mismo se cargarán las imágenes que se utilizaran en nuestro programa por medio de variables, luego tendremos una función la cual cargara los “ovnis” en la pantalla por medio de una función, luego tendremos un ciclo o array, el cual le dará movimiento a nuestros “ovnis” en pantalla. Después, nuestro código js contiene una función para crear las funciones y un “listener”, así mismo, se crea una función para la correcta función en Mozilla Firefox. Las variables declaradas en este código servirán para: cargar el Canvas, colocación de el eje x e y, la función de las teclas de disparo, la carga de la nave, la cantidad de munición, la cantidad de enemigos vivos, la velocidad en que la bala saldrá disparada, el tiempo de disparo, etc. Luego tenemos una función la cual nos permitirá disparar en el juego, así mismo la velocidad de disparo que esta tendrá, las balas de los ovnis se crearan por medio de otra función y al igual forma que la función de los disparos de la nave se podrá ajustar o cambiar su velocidad. Se deberá crear una función para la animación y los objetos de la nave, así mismo su posición y la verificación de las balas para que al momento de estas agotarse, se envié un mensaje por medio del navegador que nos dirá “Whitout Amunition (Sin municiones)”, para que volvamos a reiniciar el nivel, la creación de la función para los enemigos y sus animaciones es creada colocando un eje x e y en nuestro Canvas, ya que estos deberán de tener una animación con la cual moverse, deberemos crear una función la cual permitirá redibujar el Canvas, así mismo se creara otra función para mostrar los mensajes en pantalla, las colisiones de los enemigos también serán creadas por medio de una función que trabajara por medio de un arreglo bidimensional, tendremos la creación de la función de verificación de “WIN o GAME OVER” para verificar si el usuario gano o perdió el juego, también la función de puntaje que se incrementara de diez en diez puntos, luego tendremos una función que verificara las pulsaciones de la tecla “espacio”, la cual se encargara de disparar, así mismo la función de verificación de la munición por medio de un arreglo unidimensional, para por ultimo solo tener el refresco del Canvas y verificaciones tanto para enemigos como para la nave del usuario.

* **Programa.**

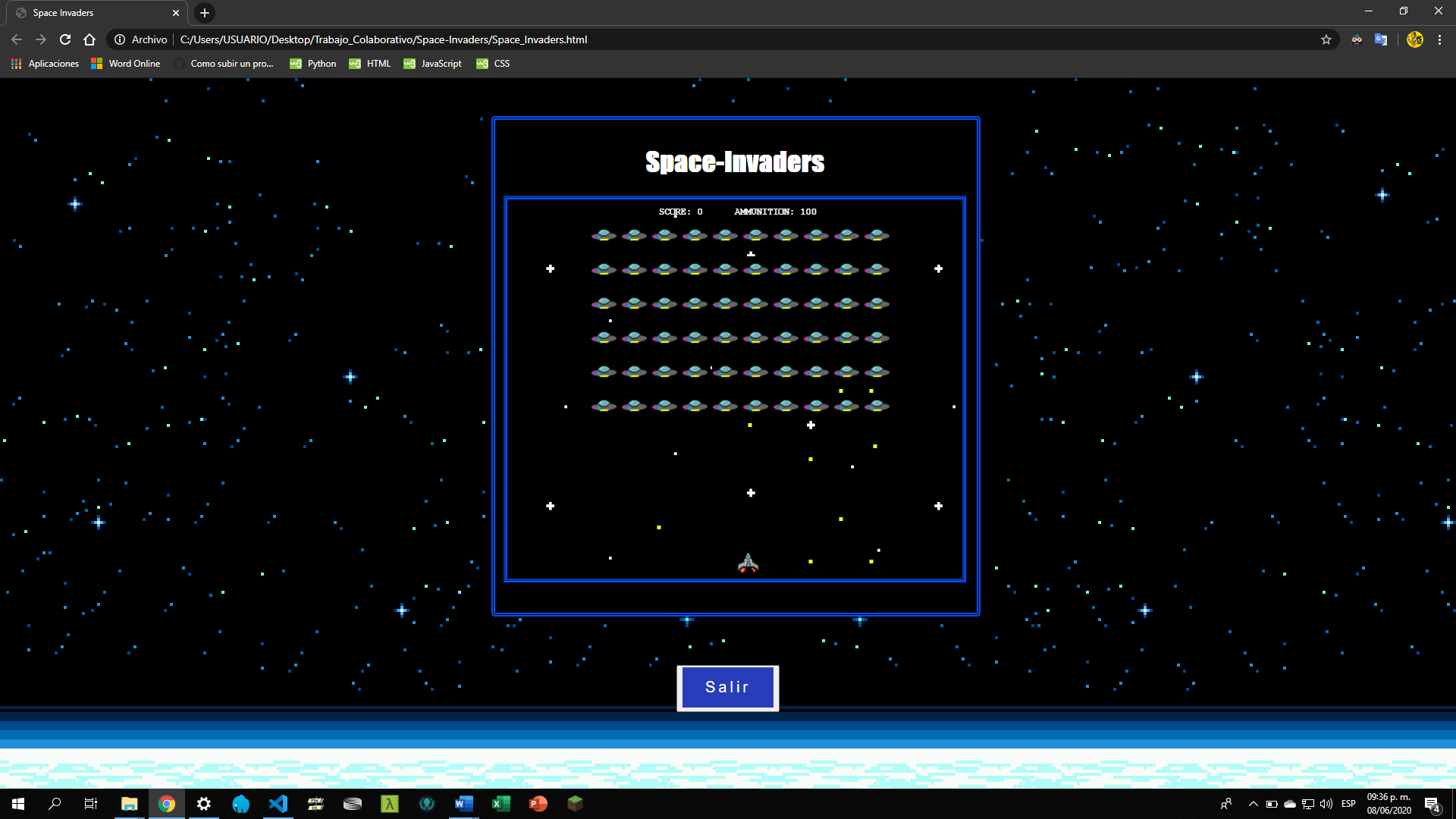


* **Manual de Usuario.**

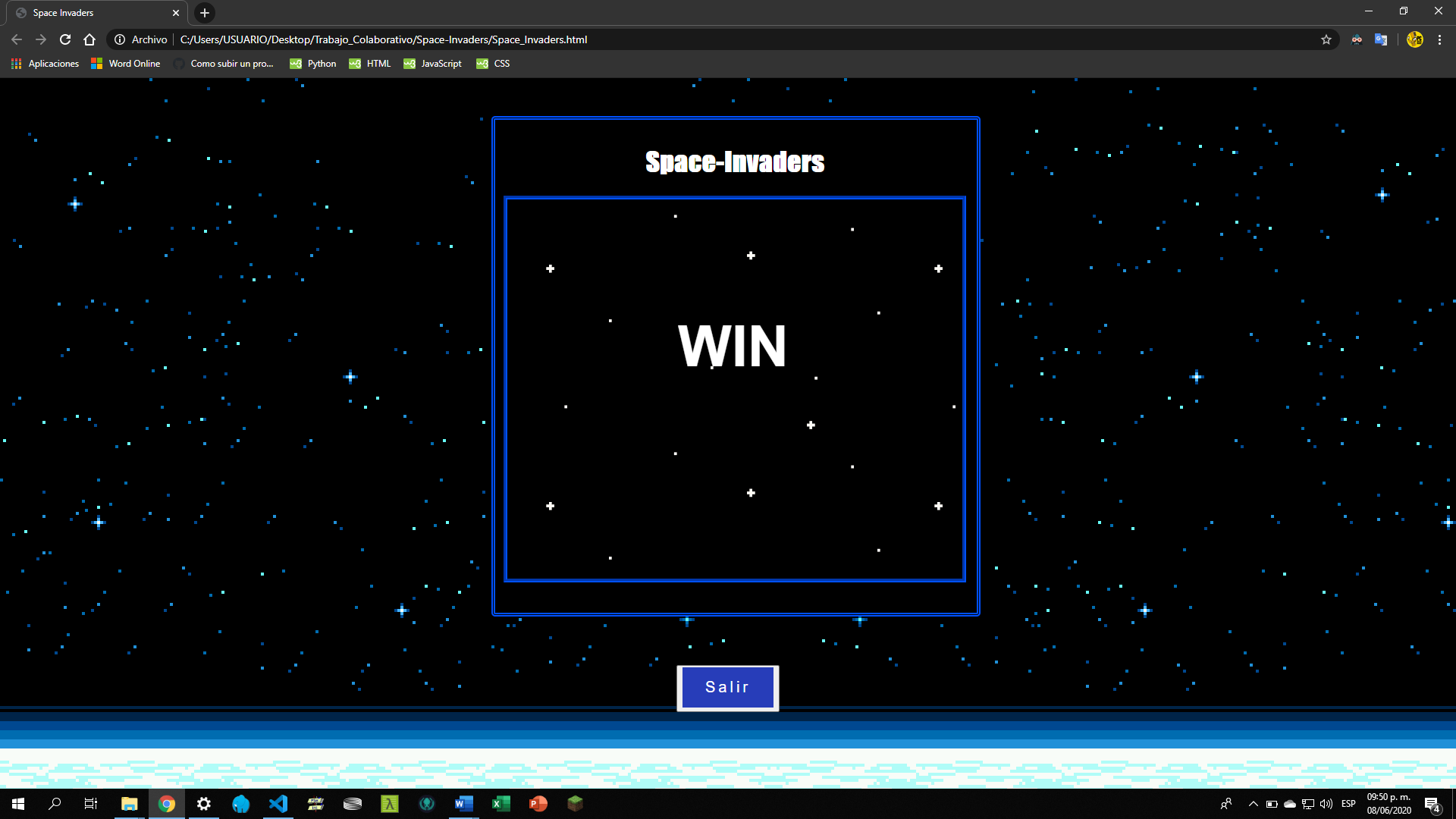
1. Al ingresar a la página se mostrará la tabla con el Canvas dentro de ella, luego deberá esperar tres segundos para poder jugar.

****

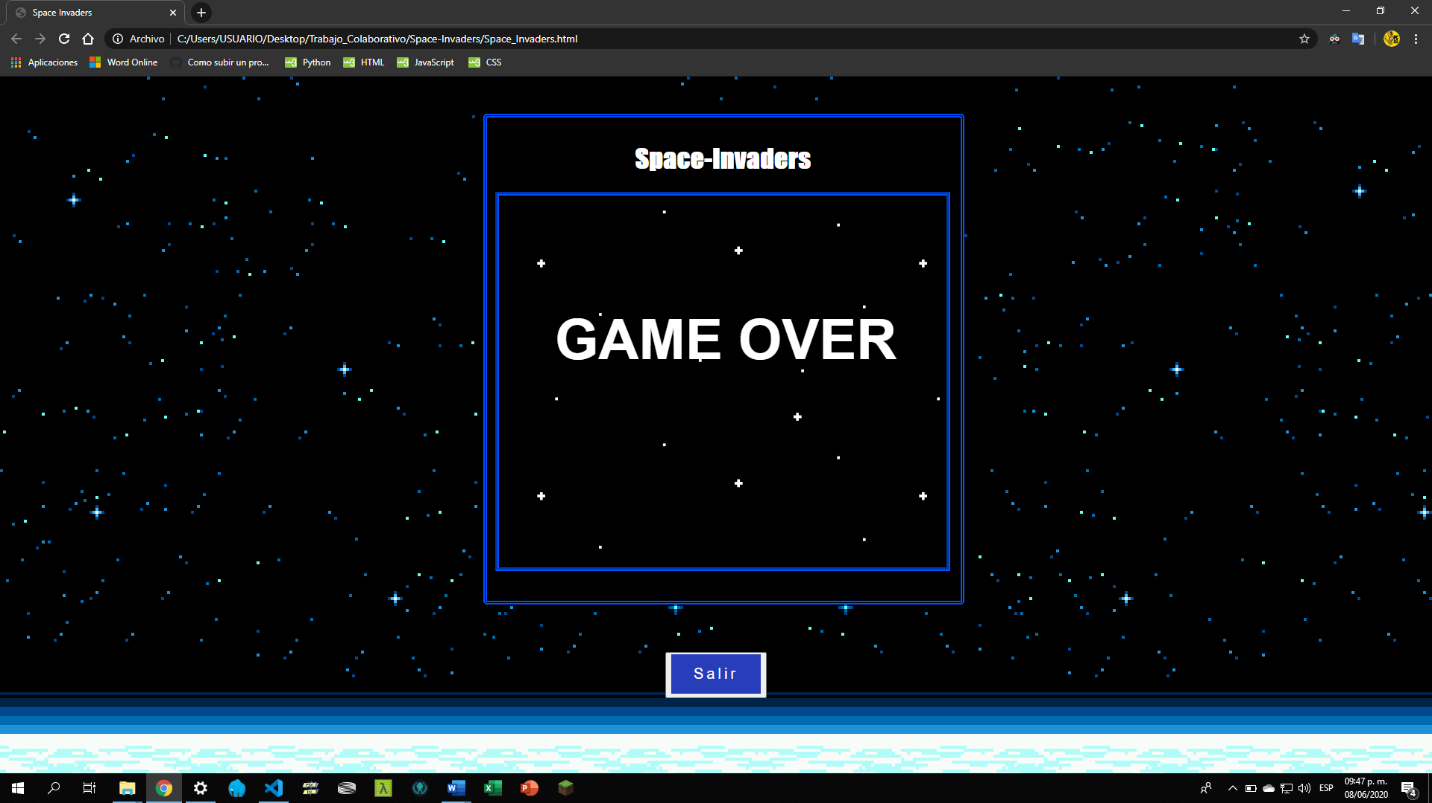
1. Al pasar los tres segundos dentro de la pagina se cargará el juego y se deberá presionar: “KEYLEFT, KEYRIGHT y la tecla SPACEBAR” para poder moverse hacia la derecha o izquierda y disparar con la barra espaciadora.



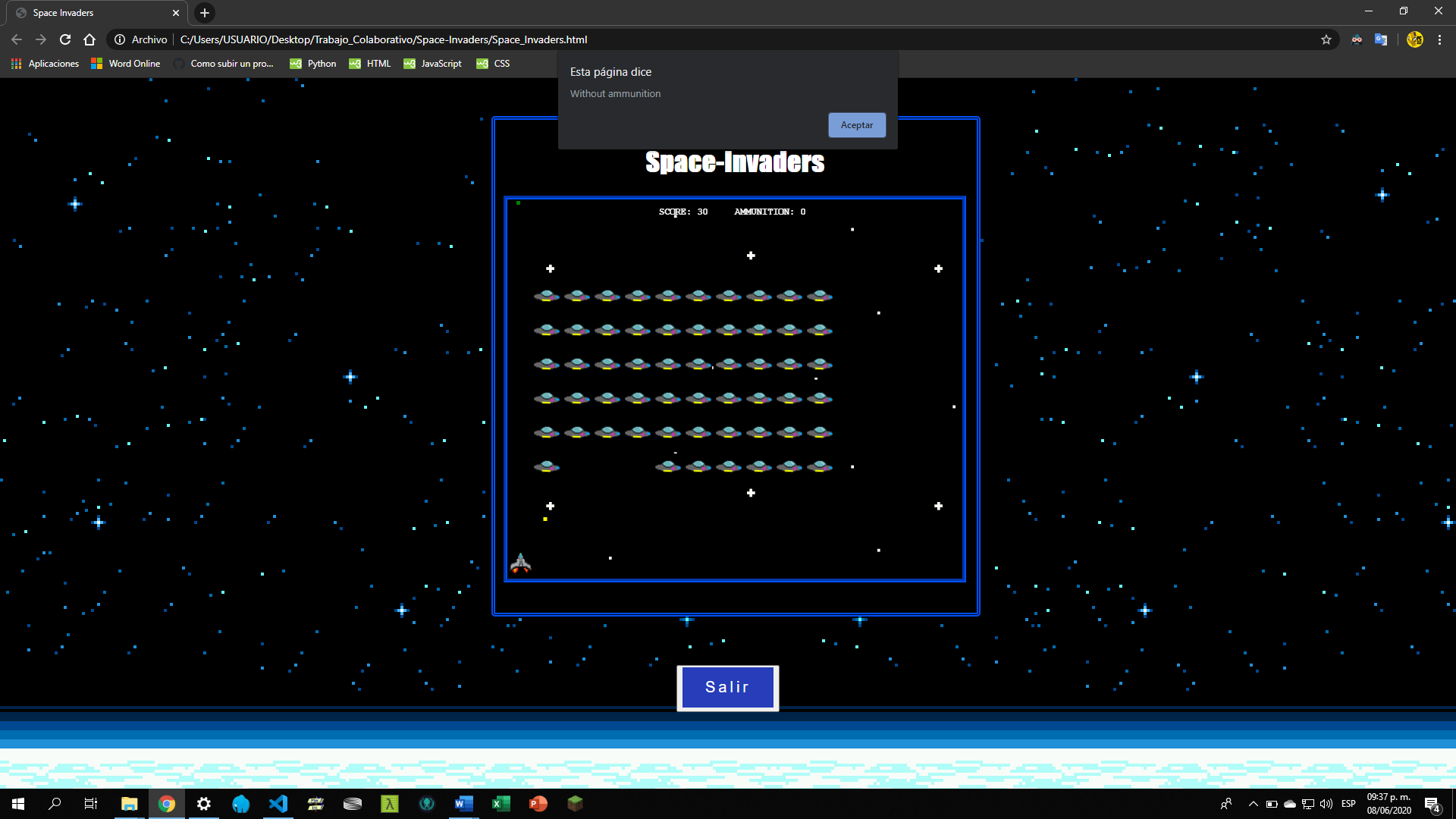
1. Este juego consta de eliminar todas las naves enemigas por medio de nuestros disparos, pero con una munición limitada y si se eliminan todas las naves se mostrará el mensaje de “WIN” en el Canvas.



1. Si una bala de los ovnis nos impacta perderemos el juego, mostrándose en el Canvas el mensaje de “GAME OVER”.

****

1. Si nos quedamos sin munición se mostrará un mensaje en nuestro navegador que dirá “Whitout Ammunition”:

****

* **Descripción del problema.**

El juego esta basado en html, js y css, por tanto se requiere del uso del método Canvas, para poder crear el dibujado y animaciones de las imágenes que contendrá el juego, así mismo se requiere de la función para poder disparar por medio de una function la cual estará ligada con la imagen de la “nave y ovni” para que se recree el efecto en las mismas, también se requiere que los “ovnis” tengan un movimiento paralelo hacia la derecha e izquierda, que disparen aleatoriamente, y que se muevan en el eje x, y e -y. La nave deberá constar de un movimiento en el eje x únicamente, desplazándose hacia la derecha e izquierda, así mismo deberá de tener la función de poder disparar. El juego deberá de contener: un marcador de puntos, un contador de munición y funciones que nos permitan saber si ganamos o perdimos el juego, así mismo, un aviso que se mostrará únicamente cuando nos quedemos sin munición para disparar.

* **Nombre del autor.**

Fernando José Martínez Medina (Fernando4268)

* **Algoritmo.**

1. Abrir el programa
2. Verificar si el Canva está cargado correctamente
3. Cargar imágenes del Canva
4. Verificar la munición de la nave
5. Verificación del cañón
6. Dibujado en x e y del Canvas
7. Verificación de los bordes o pulsaciones
8. Verificación si ganamos o perdemos el juego
9. Refrescado del Canvas

* **Diagrama de Flujo.**

**INICIO**

Abrir el programa

**SI NO**

Esta cargado el Canvas

Cargar Imágenes

**SI NO**

La nave tiene munición

Refrescado del Canvas

Pasar a la siguiente verificación

**SI NO**

Verificación del cañón

Refrescado del Canvas

Dibujado del Canvas

Pulsaciones y bordes

**SI NO**

Refrescado del Canvas

Pasar a la siguiente verificación

**SI NO**

Ganamos o perdimos

Refrescado del Canvas

**FIN**

* **Diccionario de datos.**

|  |  |  |
| --- | --- | --- |
| Termino | Tipo | Descripción |
| **HTML** | | |
| head | Metadatos | Provee información general del documento. |
| body | Contenedor | Contiene todos los aspectos de un documento. |
| table | Contenedor | Contiene celdas y columnas para la creación de tablas. |
| tr | Definición | Representa a las filas en cada tabla. |
| td | Definición | Representa a las celdas en cada tabla. |
| h1 | Identificador | Se utiliza comúnmente para las cabeceras del html. |
| button | Elemento | Representa un elemento cliqueable de tipo botón. |
| Canva | Elemento | Permite la generación de gráficos en un HTML. |
| **JavaScript** | | |
| if | Condición | Ejecuta una sentencia si una condición especificada es evaluada como verdadera en JS. |
| onload | Evento | Un evento tipo handler para el evento load event del window. |
| function | Función | La función se ejecuta cuando "algo" lo invoca o llama. |
| window | Objeto | Esta sección proporciona una pequeña referencia a todos los métodos. |
| setTimeout | Método | El método setTimeout () llama a una función o evalúa una expresión después de un número específico de milisegundos. |
| array | Objeto | Son objetos tipo-lista, cuyo prototipo tiene métodos para efectuar operaciones de recorrido y de mutación. |
| document | Objeto | Se controla la página web y todos los elementos que contiene. |
| addEventListener | Cadena tipo Evento | Registra un evento a un objeto en específico. |
| var | Sentencia | Declara una variable, opcionalmente inicializándola con un valor. |
| this | Sentencia | Atajo que refiere al objeto desde donde se invocó la función. |
| else | Condición | Si la condición es evaluada como falsa, otra sentencia puede ser ejecutada. |
| for | Condición | Crea un bucle que consiste en tres expresiones opcionales. |
| save | Método | Guarda el estado completo del Canvas añadiendo el estado actual a una pila. |
| fillText | Método | Dibuja texto relleno en el lienzo del Canvas. |
| return | Sentencia | Finaliza la ejecución de la función y especifica un valor para ser devuelto a quien llama a la función. |
| length | Propiedad | Representa la longitud de una cadena, en unidades de código UTF-16. |
| **CSS** | | |
| display | Propiedad | Es la propiedad más importante para controlar estructuras. |
| margin | Propiedad | Establece el margen para los cuatro lados. |
| background-image | Propiedad | Establece una o más imágenes de fondo para un elemento. |
| background-size | Propiedad | Especifica el tamaño de las imágenes de fondo. |
| text-align | Propiedad | Especifica la alineación horizontal del texto en un elemento. |
| font-family | Propiedad | Especifica la fuente para un elemento. |
| text-decoration | Propiedad | Especifica la decoración agregada al texto |
| font-size | Propiedad | Especifica la dimensión de la letra. |
| background | Propiedad | Especifica el color de fondo de un elemento. |
| box-shadow: | Propiedad | Adjunta una o más sombras a un elemento. |
| right | Propiedad | Afecta la posición horizontal de un elemento posicionado. |
| left | Propiedad | Afecta la posición horizontal de un elemento posicionado. |
| border | Propiedad | Permite especificar el estilo, el ancho y el color del borde de un elemento. |
| transition | Propiedad | Permiten cambiar los valores de las propiedades sin problemas, durante una duración determinada. |

* **Código fuente.**

**HTML.**

<!DOCTYPE html>

<html lang="es">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

*<!--Lectura de archivo .js y .css-->*

    <script type= "text/javascript" src= "funciones.js"></script>

    <link rel="stylesheet" a href= "estilos.css"></lin>

*<!--Creacion del titulo-->*

    <title>Space Invaders</title>

</head>

<body>

*<!--Creacion de la tabla para el Canvas-->*

    <table class= "tabla1">

        <tr>

            <td>

            <h1>Space-Invaders</h1>

*<!--Archivo Canvas para el movimiento y algunas funciones del .js-->*

            <canvas id= "miCanvas" width="600px" height= "500px">

            Tu navegador no soporta Canvas</canvas>

            </td>

        </tr>

    </table>

*<!--Creacion del boton para ir a la pagina principal-->*

    <p id= "p1">

        <button><a href="C:\Users\USUARIO\Desktop\Trabajo\_Colaborativo\Pagina Principal\index.html" class="btn-neon">

            <span id="span1"></span>

            <span id="span2"></span>

            <span id="span3"></span>

            <span id="span4"></span>

        Salir

        </a></button>

    </p>

</body>

</html>

**JavaScript.**

*//Inicio*

window.onload = function () {

    canvas = document.getElementById("miCanvas");

    if (canvas && canvas.getContext) {

        ctx = canvas.getContext("2d");

        if (ctx) {

            x = canvas.width / 2;

            (mensaje("INVADERS"));

            imgNave = new Image();

            imgOvni = new Image();

            imgOvni.src = "imagenes/ovni.png";

            imgNave.src = "imagenes/nave.png";

            imgNave.onload = function () {

                nave = new nave(0);

            }

*//Funcion para la cantidad de Ovnis en pantalla*

            imgOvni.onload = function () {

                for (var i = 0; i < 6; i++) {

                    for (var j = 0; j < 10; j++)

*//Movimiento en los ejes de los enemigos*

                    {

                        ovnis\_array.push(new Enemigo(100 + 40 \* j, 30 + 45 \* i));

                    }

                }

*//Tiempo de espera para la pantalla de carga*

                setTimeout(anima, 3000);

                disparoEnemigo = setTimeout(disparaEnemigo, tiempoDisparo);

            }

        } else {

            alert("Error");

        }

    }

}

*//Listener, Animaciones, correcta funcion de los distintos navegadores*

window.requestAnimationFrame = (function () {

    return window.requestAnimationFrame ||

        window.webkitRequestAnimationFrame ||

        window.mozRequestAnimationFrame ||

        function (*callback*) { window.setTimeout(*callback*, 20); }

})();

document.addEventListener("keydown", function (*e*) {

    teclaPulsada = *e*.keyCode;

    tecla[*e*.keyCode] = true;

});

document.addEventListener("keyup", function (*e*) {

    tecla[*e*.keyCode] = false;

});

*//Variables*

var canvas, ctx;

var x = 100;

var y = 100;

var teclaIzquierda = 37;

var teclaDerecha = 39;

var teclaEspacio = 32;

var imgNave, imgOvni;

var municion = 100;

var ultimos = new Array();

var imgAni = 0;

var imgAni2 = 0;

var enemigosVivos = 60;

var tiempoBala = true ;

var teclaPulsada = null;

var tecla = [];

var balas\_array = new Array();

var ovnis\_array = new Array();

var balasEnemigas\_array = new Array();

var endGame = false;

var disparoEnemigo;

var tiempoDisparo = 400;

var puntos = 0;

*//Balas de la nave*

function Bala(*x*, *y*, *w*) {

    this.x = *x*;

    this.y = *y*;

    this.w = *w*;

    this.dibuja = function () {

        ctx.save();

        ctx.fillStyle = "green";

        ctx.fillRect(this.x, this.y, this.w, this.w);

*//Velocidad de disparo de la nave*

        this.y = this.y - 10;

        ctx.restore();

    };

*//Balas de los Ovnis*

    this.dispara = function () {

        ctx.save();

        ctx.fillStyle = "yellow";

        ctx.fillRect(this.x, this.y, this.w, this.w);

*//Velocidad de disparo de los ovnis*

        this.y = this.y + 4;

        ctx.restore();

    };

}

*//Objetos de Nave y animaciones*

function nave(*x*) {

*//Posicion de la nave*

    this.x = *x*;

    this.y = 465;

    this.w = 30;

    this.h = 30;

    this.dibuja = function (*x*) {

        this.x = *x*;

        if(imgAni2 < 5){

            ctx.drawImage(imgNave,0   , 0   , 32  , 32  , this.x, this.y, 35  , 35);

            imgAni2 = imgAni2 + 1;

            imgAni = imgAni + 1;

            checarBalas();

*//Verificacion de balas*

        } else if(imgAni2 < 10) {

            ctx.drawImage(imgNave,32  , 0   , 32  , 32  , this.x, this.y, 35  , 35);

            imgAni2 = imgAni2 + 1;

            imgAni = imgAni + 1;

        } else{

            ctx.drawImage(imgNave,32  , 0   , 32  , 32  , this.x, this.y, 35  , 35);

            imgAni2 = 0;

        }

    };

}

*//Objetos de Enemigos y animaciones*

function Enemigo(*x*, *y*) {

    this.x = *x*;

    this.y = *y*;

    this.w = 35;

    this.veces = 0;

    this.dx = 5;

    this.ciclos = 0;

    this.num = 14;

    this.figura = true;

    this.vive = true;

    this.dibuja = function () {

*//Retraso*

        if (this.ciclos > 20) {

*//Saltos de los enemigos*

            if (this.veces > this.num) {

                this.dx \*= -1;

                this.veces = 0;

                this.num = 30;

                this.y += 40;

                this.dx = (this.dx > 0) ? this.dx++ : this.dx--;

            } else {

                this.x += this.dx;

            }

            this.veces++;

            this.ciclos = 0;

            this.figura = !this.figura;

        } else {

            this.ciclos++;

        }

        if (this.vive) {

            if (imgAni < 4) {

                ctx.drawImage(imgOvni, 0   , 0   , 32  , 32  , this.x, this.y, 35  , 35);

*//Dibujado de: imgfile, xini, yini, wimg, himg, xpos  , ypos  , wrez, hrez*

            } else if(imgAni < 8) {

                ctx.drawImage(imgOvni, 32, 0, 32, 32, this.x, this.y, 35, 35);

            } else if(imgAni < 12) {

                ctx.drawImage(imgOvni, 64, 0, 32, 32, this.x, this.y, 35, 35);

            } else if(imgAni > 11) {

                ctx.drawImage(imgOvni, 0, 0, 32, 32, this.x, this.y, 35, 35);

                imgAni = 0;

            }

        } else {

            ctx.fillStyle = "black";

            ctx.fillRect(this.x, this.y, 35, 30);

        }

    };

}

*//Redibujado de Canva y funciones*

function anima() {

    if (endGame == false) {

        requestAnimationFrame(anima);

        verifica();

        pinta();

        colisiones();

    }

}

*//Mensajes en pantalla*

function mensaje(*cadena*) {

    var lon = (canvas.width - (50 \* *cadena*.length)) / 2;

    ctx.fillStyle = "white";

    ctx.clearRect(0, 0, canvas.width, canvas.height);

    ctx.font = "bold 75px Arial";

    ctx.fillText(*cadena*, lon, 220);

}

*//Colisiones de enemigos y nave*

function colisiones() {

    for (var i = 0; i < ovnis\_array.length; i++) {

        for (var j = 0; j < balas\_array.length; j++) {

            enemigo = ovnis\_array[i];

            bala = balas\_array[j];

            if (enemigo != null && bala != null) {

                if ((bala.x > enemigo.x) &&

                    (bala.x < enemigo.x + enemigo.w) &&

                    (bala.y > enemigo.y) &&

                    (bala.y < enemigo.y + enemigo.w)) {

                    enemigo.vive = false;

                    enemigosVivos = enemigosVivos - 1;

                    ovnis\_array[i] = null;

                    balas\_array[j] = null;

                    puntos += 10;

                    score();

                }

            }

        }

    }

    for (var j = 0; j < balasEnemigas\_array.length; j++) {

        bala = balasEnemigas\_array[j];

        if (bala != null) {

            if ((bala.x > nave.x) &&

                (bala.x < nave.x + nave.w) &&

                (bala.y > nave.y) &&

                (bala.y < nave.y + nave.h)) {

                gameOver();

            }

        }

    }

}

*//Verificacion si ganamos o perdemos el juego*

function gameOver() {

    ctx.clearRect(0, 0, canvas.width, canvas.height);

    balas\_array = [];

    ovnis\_array = [];

    balasEnemigas\_array = [];

    if( enemigosVivos == 0 ){

        mensaje("WIN");

    }else{

        mensaje("GAME OVER");

    }

    endGame = false;

    clearTimeout(disparoEnemigo);

}

*//Funcion del puntaje*

function score() {

    ctx.save();

    ctx.fillStyle = "white";

    ctx.clearRect(0, 0, canvas.width, 20);

    ctx.font = "bold 12px Courier";

    ctx.fillText("SCORE: " + puntos, 200, 20);

    ctx.restore();

}

*//Funcion de las municiones*

function municiones() {

    ctx.save();

    ctx.fillStyle = "white";

    ctx.clearRect(0, 20, canvas.width, 20);

    ctx.font = "bold 12px Courier";

    ctx.fillText("AMMUNITION: " + municion, 300, 20);

    ctx.restore();

}

*//Verificacion de los bordes y pulsaciones de tecla espacio para disparos*

function verifica() {

    if (tecla[teclaDerecha]) x += 5;

    if (tecla[teclaIzquierda]) x -= 5;

*//Verificacion del cañon*

    if (x > canvas.width - 20) x = canvas.width - 20;

    if (x < 0) x = 0;

*//Disparo*

    if (tecla[teclaEspacio]) {

        if (tiempoBala == true && municion !=0 ){

            tiempoBala = false;

            balas\_array.push(new Bala(nave.x + 12, nave.y - 3, 5));

            (municion >0)?municion = municion - 1 : false;

            tecla[teclaEspacio] = false;

            disparaEnemigo();

*/\*Tiempo de disparo\*/*

            setTimeout(function(){tiempoBala = true;},150);

        }

    }

}

*//Verificacion de la municion*

function checarBalas(){

    var balasArrayVal = 0;

    for(let i = 0 ; i < balas\_array.length; i++){

        if(balas\_array[i] != null){

            balasArrayVal = 1;

        }

    }

    if(municion == 0 && balas\_array.length == 100 && balasArrayVal == 0 && enemigosVivos > 0){

        tecla[teclaEspacio] = false;

            alert("Without ammunition");

            gameOver();

    }

}

*//Refresh de Canva*

function pinta() {

    ctx.clearRect(0, 0, canvas.width, canvas.height);

    score();

    municiones();

    nave.dibuja(x);

*//Balas*

    for (var i = 0; i < 100; i++) {

        if (balas\_array[i] != null) {

            balas\_array[i].dibuja();

            if (balas\_array[i].y < 0) balas\_array[i] = null;

        }

    }

*//Balas enemigas*

    for (var i = 0; i < balasEnemigas\_array.length; i++) {

        if (balasEnemigas\_array[i] != null) {

            balasEnemigas\_array[i].dispara();

            if (balasEnemigas\_array[i].y > canvas.height) balasEnemigas\_array[i] = null;

        }

    }

*//Enemigos*

    numEnemigos = 0;

    for (var i = 0; i < ovnis\_array.length; i++) {

        if (ovnis\_array[i] != null) {

            ovnis\_array[i].dibuja();

            if (ovnis\_array[i].y == nave.y) {

                gameOver();

            }

            numEnemigos++;

        }

    }

    if (numEnemigos == 0) gameOver();

}

*//Control de enemigos que disparan*

function disparaEnemigo() {

    for (var i = ovnis\_array.length - 1; i > 0; i--) {

        if (ovnis\_array[i] != null) {

            ultimos.push(i);

        }

        if (ultimos.length >= 15) break;

    }

    Array.prototype.clean = function(*deleteValue*) {

        for (var i = 0; i < this.length; i++) {

                if (this[i] == *deleteValue*) {

                    this.splice(i, 1); i--;

                }

            } return this;

        };

    ovnis\_array.clean(undefined);

    d = ultimos[Math.floor(Math.random() \* ovnis\_array.length)];

    if(ovnis\_array[d] == null || d == null){

        ovnis\_array.clean(undefined);

        d = Math.floor(Math.random() \* ovnis\_array.length);

    }

    balasEnemigas\_array.push(new Bala(ovnis\_array[d].x + ovnis\_array[d].w / 2, ovnis\_array[d].y, 5));

    clearTimeout(disparoEnemigo);

    disparoEnemigo = setTimeout(disparaEnemigo, tiempoDisparo);

}

**CSS.**

*/\*Ajustes del body\*/*

body {

    display: block;

    width: 630px;

    margin: 0 auto;

    background-image: url("C:/Users/USUARIO/Desktop/Trabajo\_Colaborativo/img/space.jpg");

    background-size: 100%;

    align-content: center;

}

*/\*Creacion de los estilos para el h1\*/*

h1 {

    text-align: center;

    font-family: fantasy;

    font-size: 35px;

    color: white;

}

*/\*Creacion de los estilos para el h4\*/*

h4 {

    font-family: arial;

    font-size: medium;

    color: white;

    font-size: 12px;

}

*/\*Creacion del estilo para la tabla\*/*

.tabla1 {

    display: block;

    width: 645px;

    height: 660px;

    margin: 3px;

    border: 5px;

    background-color: black;

    border-spacing: 10px 10px;

    margin-top: 50px;

    border-collapse: separate;

    border-style: double;

    border-radius: 5px;

    border-color: #0051ff;

}

*/\*Creacion del estilo para p\*/*

p{

    display: flex;

    justify-content: center;

    align-items: center;

    height: 150px;

    width: center;

    background-size: auto;

    font-family: consolas;

    margin: 20px;

    }

    .btn-neon{

    position: relative;

    display: inline-block;

    padding: 15px 30px;

    color: white;

    letter-spacing: 4px;

    text-decoration: none;

    font-size: 20px;

    overflow: hidden;

    transition: 0.5s;

    background-color: rgb(39, 61, 185);

    }

    .btn-neon:hover{

     background: gray;

     box-shadow: 0 0 10px gray, 0 0 40px gray, 0 0 80px gray;

     transition-delay: 1s;

     }

    .btn-neon span{

     position: absolute;

     display: block;

            }

     #span1{

     top: 0;

     left: -100%;

     width: 100%;

     height: 2px;

     background: linear-gradient(90deg, transparent,#a945c7);

     }

     .btn-neon:hover #span1{

     left: 100%;

     transition: 0.5s;

     }

     #span3{

     bottom: 0;

     right: -100%;

     width: 100%;

     height: 2px;

     background: linear-gradient(270deg, transparent,#a945c7);

     }

     .btn-neon:hover #span3{

      right: 100%;

      transition: 1s;

      transition-delay: 0.5s;

      }

      #span2{

      top: -100%;

      right: 0;

      width: 2px;

      height: 100%;

      background: linear-gradient(180deg, transparent,#a945c7);

      }

      .btn-neon:hover #span2{

      top: 100%;

      transition: 1s;

      transition-delay: 0.25s;

      }

      #span4{

      bottom: -100%;

      left: 0;

      width: 2px;

      height: 100%;

      background: linear-gradient(360deg, transparent,#a945c7);

      }

      .btn-neon:hover #span4{

      bottom: 100%;

      transition: 1s;

      transition-delay: 0.75s;

      }

*/\*Creacion de el tamaño, fondo y la sombra de la caja del Canvas\*/*

#miCanvas {

    border: groove 5px #0051ff;

    background-image: url("imagenes/universo.jpg");

    box-shadow: 7px 7px 5px black;

}