Nombres:

- Luis Oña
- Mateo Salvador

Materia: Aplicaciones Web y Móviles - Gr1

Fecha de entrega: 02/03/2022

Tema: Juego con JavaScript y HTML

I. Objetivos

a) Aplicar sentencias condicionales y repetitivas en JavaScript.
b) Definir funciones que permitan identificar los procesos que se ejecutarán en JavaScript y podrán ser visibles a través de un documento HTML

II. Instrucciones

Para el código desarrollado en clase, agregar las siguientes funcionales:

1. Desaparecer un enemigo cuando le alcance un misil
2. Agregar un panel de score en la parte superior en la que se muestre el número de enemigos destruidos.
3. Agregar al panel de score el número de vidas del player (5 vidas)
4. El juego termina cuando el player ya no tiene vidas.

III. Desarrollo

En el código desarrollado en clases, se logro implementar el movimiento del avión del jugador y los enemigos, de la misma manera se definieron las restricciones respectivas en cuanto a la movilidad a través del background del océano.

Como primer paso se definió un estilo en el apartado <style> en donde se mostrara el score panel de la siguiente manera

```
.scorePanel{
        background-color: lightgreen;
        width: 250px;
        height: 900px;
        float: left;
        font-family: 'Courier New', Courier, monospace;
        font-size: 0.75cm;
        font-weight: bold;
        color:black;
    }
```

Se definen las características de color de fondo (background-color), ancho(width), alto(height), propiedad de posicionamiento (float), propiedad de familia de fuente (Font-family), tamaño de fuente (Font-size), peso de fuente (Font-weight), color de texto (black).

En el apartado <body> se definio una nueva división la cual corresponde al scorePanel la cual se definió de la siguiente manera

```
<div id="Panel"></div>
```

Esta división se lo realiza aparte de la división existente con el background del océano, debido a que se mostrara al lado izquierdo del mismo, pero en diferente división como se observara más adelante.

En el apartado de <script>, se procede a crear algunas variables que permitirán el almacenamiento de algunos valores que serán necesarios para mostrarlos en el score Panel.

```
var lives=5;
var score=0;
var lastScore=0;
var velocidad=225;
var highScore=0;
var scores=[highScore];
```

En las indicaciones se establece que el número de vidas debe ser 5, como contadores las variables serán score, lastScore y highScore , se establece la variable de la velocidad del juego y una variable scores como un vector que ira almacenando las mayores puntuaciones.

En el mismo apartado de <script>, se establecieron algunas funciones que permitirán cumplir con las instrucciones planteadas y las cuales se detallan a continuación.

- Función de ScoreAndLives: Servirá para mostrar en el panel los valores que están almacenando los contadores de puntaje, vidas, ultimo puntaje y puntaje mas alto.

```
function ScoreAndLives(){
            var content = "";
            content = "<div class='scorePanel' style= 'left: "+ 0 + "px; top: " + 0 +
"px'><h2 style='text-align: center'>AIR COMBAT</h2>Score: " + score +
            "</br>Lives: "+ lives +"</br>Last Score: "+ lastScore + "</br>High Score: "+
highScore +"</div>";
            console.log(content);
            document.getElementById("Panel").innerHTML = content;
        }
```

- Funcion de MoveEnemies: La misma ya estaba funcionando, sin embargo se realizaron modificaciones de tal manera que cuando un enemigo desaparezca del mapa, inmediatamente aparezca uno nuevo en una posición randomica, para esto se utilizaron los métodos splice y push, en donde el primero elimina elementos existentes en un array y el segundo permite agregar elementos al final de un array, de esta manera se complementan la eliminación y agregación.

```
  function moveEnemies(){
            for(var i=0; i<enemies.length; i++){
                if(enemies[i].top < 815){
                enemies[i].top +=10;
                }
                else{
                enemies.splice(i,1)
                enemies.push({left: Math.floor(Math.random()*501),top:0})
                }
            }
        }
```

- Funcion KillerShot: Se trata de la función que hará que los enemigos desaparezcan una vez que han sido alcanzados por los misiles, la misma se compone de dos blucles for los cuales

se ejecutaran respectivamente para los enemigos y los misiles y cumplirán la condición de alcance de un misil al enemigo en algún punto del mapa, si esto sucede, se eliminara el misil generado junto con el enemigo que lo recibió, de la misma manera por cada enemigo eliminado se agrega uno nuevo en un lugar aleatorio y por ultimo el contador del puntaje se ira incrementando de uno en uno

```
  function killershot(){
            for(var i=0; i<enemies.length; i++){
                for(var j=0; j<missile.length;j++){
                    if((missile[j].left-35)-enemies[i].left<=15 &&
missile[j].top-enemies[i].top<=15){
                        missile.splice(j,1);
                        enemies.splice(i,1);
                        enemies.push({left:
Math.floor(Math.random()*501),top:0});
                        score ++;
                        break;
                    }
                }
            }
        }
```

- Funcion Colition: Se define una esta función con la idea de que si un enemigo golpea al jugador, el enemigo se elimine y vuelva a aparecer uno nuevo en el mapa y descuente en 1 la vida del jugador, tiene una lógica similar al anterior código con la diferencia de que en este no se toma en cuenta el misil.

```
 function colition(){
            for(var i=0; i<enemies.length; i++){
                    if((player.left)-enemies[i].left<=Math.abs(15) &&
player.top-enemies[i].top<=Math.abs(15)){
                        enemies.splice(i,1);
                        enemies.push({left:
Math.floor(Math.random()*501),top:0})
                        lives --;
                        break;
                }
            }
        }
```

- Función gameover: Se crea esta función con la finalidad de establecer un control en cuanto al contador de vidas en donde si se cumple que el jugador tiene un total de cero vidas, se procedera a enviar una alerta de fin de juego y se mostrara el puntaje obtenido, se almacenara este como ultimo puntaje y se llamara a la Función highscore en el caso de que el mismo sea el mas alto registrado una vez realizado este proceso se reinicia las vidas y el puntaje para un nuevo juego.

- Funcion gameover

```
function gameover(){
            if(lives==0){
                alert("GAME OVER!! Tu puntaje fue: "+ score +"
Aplasta OK para reiniciar");
                lastScore=score;
```

```
            scores.push(score);
            highscore();
            lives=5;
            score=0;
            player.top=720;
            player.left=400;
        }
    }
```

- Función highscore

```
function highscore(){
        for(var i=0; i<scores.length; i++){
            if(highScore < scores[i])
            highScore = scores[i];
        }
    }
```

- Funcion dificultad: Se define una función en la cual se cumple que cada vez que un jugador llegue a un multiplo de 5 en el puntaje y la velocidad sea mayor a 160, la velocidad del juego ira decrementándose en 10

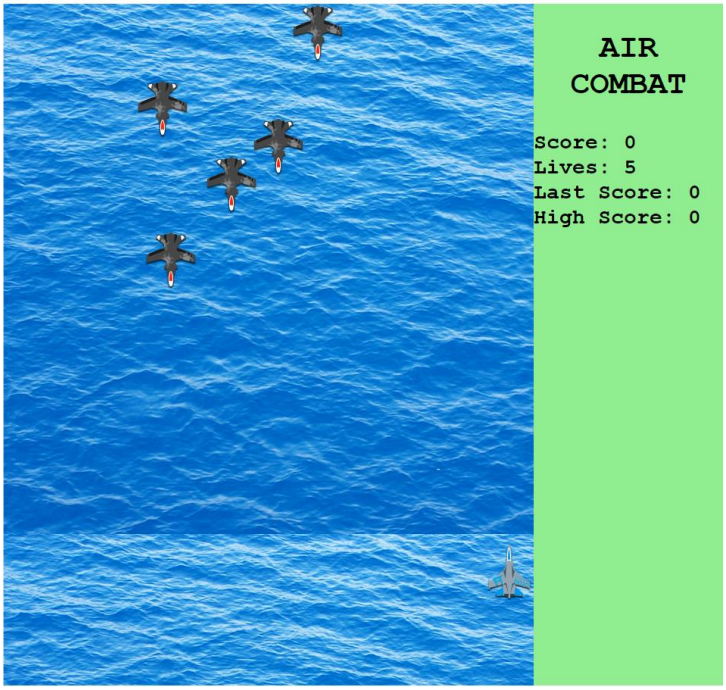
```
function dificultad(){
        if(score % 5 ==0 && velocidad > 160){
            velocidad-=10;
            console.log(velocidad)
        }
    }
```

- Funciones gameLoop y missileLoop: Son funciones que contienen a las funciones creadas con anterioridad, son las encargadas de permitir que el programa se siga ejecutando continuamente según las condiciones generadas.

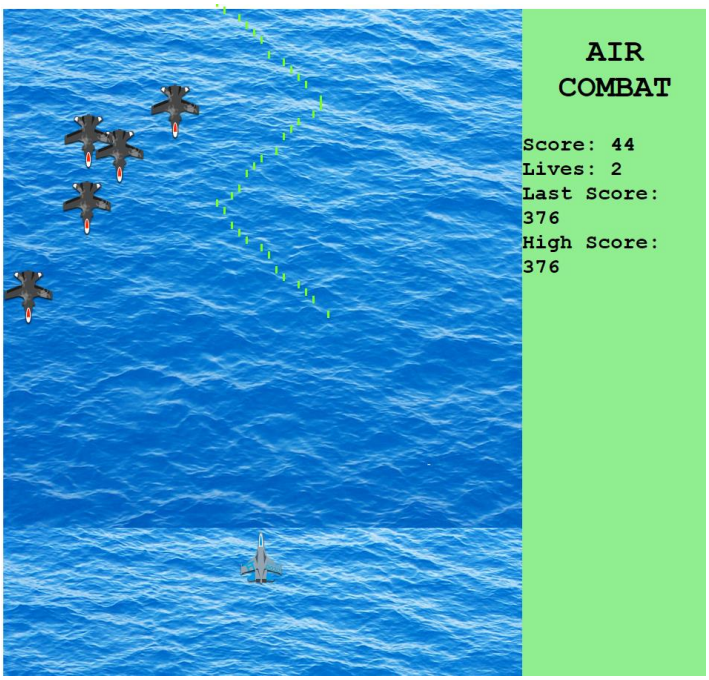
```
function gameloop(){
        ScoreAndLives();
        drawPlayer();
        drawEnemy();
        moveEnemies();
        colition();
        dificultad();
        gameover();
        setTimeout(gameloop,velocidad);
    };
    function missileloop(){
        drawMissile();
        moveMissiles();
        killershot();
        setTimeout(missileloop,150);
    }
    missileloop();
    gameloop();
```

- Evidencias de ejecución

1. Inicio



2. Jugabilidad



3. Finalización