EN QUÉ CONSISTE EL EXAMEN

El examen está dividido en dos partes que coinciden con lo que hemos trabajado en clase (ambos trimestres).
Deberás realizar las dos partes.
Para aprobar el examen deberás sacar más de 2,5 puntos en cada una de las partes.

CRITERIOS DE CORRECCIÓN DEL EXAMEN

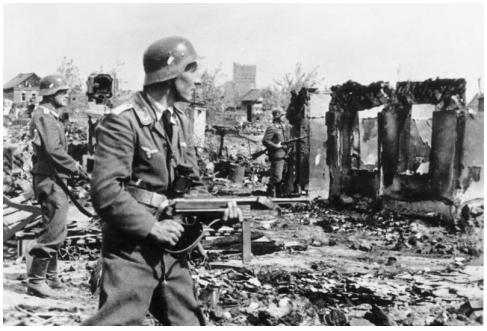
Parte I. Ejercicio Javascript (5 puntos)

Parte II. Ejercicio ReactJS (5 puntos)

PARTE I. STALINGRADO

Leningrado fue según los historiadores el evento más horrible que sucedió en la Segunda Guerra Mundial y uno de los peores de la historia. Tuvo una duración de unos novecientos días y las muertes se cuentan por cientos de miles.

Aproximadamente se estima que murieron “de hambre” ochocientos mil personas lo cual supuso un tercio de la población total de la ciudad. La gente empezó a mezclar tierra con harina y se sucedieron numerosos episodios de canibalismo entre los habitantes de la ciudad.



Según dijo Adolf Hitler "Leningrado debe ser borrado de la faz de la tierra. No nos interesa en absoluto salvar civiles".

Crea una matriz de 9x9 en la que aparezcan A(alemanes) y R(rusos) simulando el campo de batalla.

Los rusos quieren dejar a los alemanes aislados. Los rusos intentarán aislar a los alemanes en sus respectivas zonas haciendo una especie de barrera. Si los alemanes están aislados pierden, sino pierden los rusos.
Tienes que tener en cuenta que en diagonal las tropas no pueden avanzar. Las tropas pueden avanzar en horizontal y vertical.
Realiza un programa que averigua qué bando gana.

Imagina que la matriz proporcionada es la siguiente:

A	A			R				
A	A			R				
A	A			R				
				R				
				R				
				R			A	A
				R			A	A
				R			A	A
				R			A	A

El programa tras procesarla mostrará lo siguiente:

¡GANAN LOS RUSOS!

A	A			R				
A	A			R				
A	A			R				
				R				
				R				
				R			A	A
							A	A
				R			A	A
				R			A	A

El programa tras procesarla mostrará lo siguiente

¡GANAN LOS ALEMANES!

Realiza el programa teniendo en cuenta que el campo de batalla puede ser un 9x9 u otro tamaño.

Imagen: **Stalingrad, Luftwaffen-Soldaten in Ruinen. Bundesarchiv, Bild 183-B22478 / Rothkopf / CC-BY-SA 3.0**

Te adjunto parte de mi código que tendrás que utilizar para terminar el programa:

```
<!DOCTYPE html>
<html>
<body>

<h1>STALINGRADO</h1>
<p id="matriz"></p>
<h2 id="ganador"></h2>

<script>
let battlefield = [
["A","A","","","R","","","","",],
["A","A","","","R","","","","",],
["","","","","R","","","","",],
["","","","","R","","","","",],
```

```
["","","","","R","","","","",""],
["","","","","R","","","","",""],
["","","","","R","","","","",""],
["","","","","R","","","A","A"],
["","","","","R","","","A","A"]
];

</script>

</body>
</html>
```

Tienes que resolver el programa anterior utilizando recursividad. Cosas que se valorarán para la corrección:

- Legibilidad del código.
- Longitud del código.
- No se valorará el aspecto estético de la web.
- El programa realiza la funcionalidad requerida.
- Que no haya errores en la consola.
- La eficiencia.
- La modularidad (alta cohesión y bajo acoplamiento).
- Que el código sea limpio, flexible, reutilizable y mantenible.
- Sigue los principios SOLID.
- Otros criterios.

PARTE II. REACTJS. REQUISITOS DE LA APLICACIÓN

Se pide que crees un programa ReactJS que muestre los últimos 2 botones pulsados de X botones que tendrá tu aplicación.

El programa tendrá el siguiente aspecto:

De momento no se ha pulsado ningún botón.
Si pulso el primero:

Si pulso el segundo:

Si pulso el tercero:

Si pulso el último:

Como ves, el programa solo recuerda los dos últimos botones pulsados.

Muy importante: Tu programa tendrá dos componentes, App y Botoncillo.

Te paso parte de mi código **que tendrás que utilizar para terminar la aplicación**:

```
class App extends Component{
 constructor(props) {
   super(props);
   this.state = {
      cuantos: 5, // mi aplicación mostrará 5 botones pero si cambio este número a 25 aparecerán 25 botones. Se supone que más de 99 no serán necesarios.
      // DEFINE AQUÍ TU ESTADO
    };
// En el constructor puedes inicializar el estado si lo deseas
 }


      // SEGURAMENTE NECESITARÁS IMPLEMENTAR MÁS DE UN MÉTODO.


/*IMPORTANTE: SE PIDE UNA RENDERIZACIÓN DINÁMICA. TEN EN CUENTA QUE TIENES 5 BOTONES PERO PODRÍAS TENER 25, 7  o LOS QUE SEA*/
 render(){
   return (
     <div className="App">
       <header className="App-header">
         // RENDERIZA AQUÍ LO QUE NECESITES
       </header>
     </div>
   );
 }

}
```

```
function Botoncillo(props){
 return (
   // MUESTRA AQUÍ EL BOTÓN CON EL COLOR
CORRESPONDIENTE
 );
}
```

Tu programa deberá realizar correctamente lo siguiente:

1. Crear los botones dinámicamente.
2. Mostrar de otro color los 2 últimos botones pulsados. Al principio todos son grises (secondary)
3. Se considerará no válido un programa que solo tenga un componente funcional. No puedes tener la lógica solo en un componente, tienes que utilizar callbacks y props.
4. Tienes que utilizar el código anterior obligatoriamente.

En mi programa como he utilizado bootstrap he definido el color de los botones como “secondary”/no pulsado y “danger”/pulsado

Al igual que en el ejercicio anterior, se valorarán para la corrección:

- Legibilidad del código.
- Longitud del código.
- No se valorará el aspecto estético de la web.
- El programa realiza la funcionalidad requerida.
- Que no haya errores en la consola.
- La eficiencia.
- La modularidad (alta cohesión y bajo acoplamiento).
- Que el código sea limpio, flexible, reutilizable y mantenible.
- Sigue los principios SOLID.
- Otros criterios.

Si quieres utilizar BootStrap /reactstrap

Adding Bootstrap

Install reactstrap and Bootstrap from NPM. Reactstrap does not include Bootstrap CSS so this needs to be installed as well:

```
npm install --save bootstrap
```

```
npm install --save reactstrap react react-dom
```

Import Bootstrap CSS in the `src/index.js` file:

```
import 'bootstrap/dist/css/bootstrap.min.css';
```

Import required reactstrap components within `src/App.js` file or your custom component files:

```
import { Button } from 'reactstrap';
```

Now you are ready to use the imported reactstrap components within your component hierarchy defined in the render method. Here is an example of buttons using reactstrap.


```
    <Button color="primary">primary</Button>{' '}
    <Button color="secondary">secondary</Button>{' '}
    <Button color="success">success</Button>{' '}
    <Button color="info">info</Button>{' '}
    <Button color="warning">warning</Button>{' '}
    <Button color="danger">danger</Button>{' '}
    <Button color="link">link</Button>
```

Algunos métodos de los componentes react interesantes.

Estos métodos se llaman cuando se crea una instancia de un componente y se inserta en el DOM:

constructor()
render()
componentDidMount()
componentWillMount() Este último está obsoleto(deprecado) y te dicen que utilices el constructor. Te va a seguir funcionando pero te dará un warning (pero funciona)..