Blue Tech Test - Examen de conocimientos para desarrollar en Cells

- **Número de preguntas:** 25
- **Tiempo:** 2 horas 30 minutos
- **Número de secciones**: 3
- Programación Orientada a Objetos
- 5 Preguntas - Fallos permitidos: 1
- JavaScript
- 10 Preguntas - Fallos permitidos: 2
- Polymer 2.
- 10 Preguntas - Fallos permitidos: 3
- **Tipo de examen:** Opción multiple
- *Para obtener una nota aprobatoria de todo el examen debes aprobar cada una de las secciones, lo que implica que si una de las secciones rebasa su número máximo permitido de fallos automáticamente habrás reprobado el examen independientemente del hecho de que las secciones restantes no tengan error alguno.*
- **El examen se ejecuta en pantalla completa de la cual no debes salir hasta terminar el examen, los cambios de pestaña o ventana son registrados en la plataforma así como la apertura de la consola de desarrollador, si dichos eventos son registrados el examen será anulado y por ende**

no acreditado.

Sección I - POO

Buenas prácticas

1. La Alta Cohesión deberia se acompañada de…

- **Polymorfismo**
- **Instancias**
- **Bajo acoplamiento**
- **Patrón de diseño Factory method**

Herencia

2. ¿Cuál de las siguientes afirmaciones se corresponde con el concepto de Herencia en POO?

- **Evita que otros objetos modifiquen o tengan acceso a las propiedades encapsuladas de un objeto.**
- **La clase Empleado es la super clase de las clases, Empleado Sindicalizado, Empleado de Confianza, Empleado Externo**
-

Las funciones se resuelven en tiempo de ejecución en lugar de tiempo de compilación

- [ ] **Al acoplamiento y baja cohesión son el resultado de una jeraquia de clases bien definidas.**

Padres e hijos

3. Cuando dos o mas clases sirven como clase base para una clase derivada, la situación se conoce como

- [ ] **Polimorfismo**
- [x] **Herencia múltiple**
- [ ] **Encapsulamiento**
- [ ] **Herencia jerárquica**
- [ ] **Ninguna de las anteriores**

Responsabilidades

4. La información que almacena una clase debe ser coherente y estar relacionada lo mas posible con la clase misma. ¿Estamos hablando de?

- [x] **Alta cohesion**
- [ ] **Especialización**
- [ ] **Monomorfismo**
- [ ] **Bajo acoplamiento**

Polimorfismo

5. ¿Cuál de las siguientes afirmaciones es acerca del polimorfismo en POO?

- **Es el ocultamiento del estado de un objeto**
- **Es la capacidad de suministrar datos al código**
- **Es la capacidad que tienen los objetos de una clase de responder a un evento en función de los parámetros utilizados en su invocación**
- **Es cuando un objeto adquiere todas las propiedades de otro objeto**

6. ¿Cuál es el modelo para creación de objetos?

- **Lista**
- **Instancia**
- **Array**
- **Clase**

7. ¿Cuál de las siguientes afirmaciones se refiere al polimorfismo?

- **Ocultamiento del estado de un objeto**
- **Capacidad de suministrar datos al codigo**
-

Capacidad que tienen los objetos al responder a un evento en funcion de los parametros utilizados en su invocacion

- **Cuando un objeto adquiere todas las propiedades de otro**

8. Es el mecanismo que permite crear nuevas clases a partir de clases existentes extendiendo y refinando sus capacidades

- **Polimorfismo**
- **Encapsulamiento**
- **Diseño de Patrones**
- **Herencia**

9. ¿A que se refiere cuando la información de una clase debe ser coherente y relacionada con la misma?

- **Alta cohesion**
- **Especialización**
- **Bajo acoplamiento**

10. ¿Cuál es la diferencia entre un objeo y una clase?

- **Un objeto es una instancia de una clase.**
- **Un objeto genera funciones, valores para ser usados por una clase.**

11. A que se refiere cuando una clase es derivada de otra?

- **Herencia multiple**

- [x] **Herencia jerarquica**

12. Metodo especial de la clase que permite inicializar los valores de un objeto

- [ ] **Objetos**
- [x] **Constructor**
- [ ] **Herencia**
- [ ] **Virtual**

13. ¿A que se refiere cuando los detalles internos de una clase pueden ser ocultados del exterior?

- [x] **Encapsulamiento**

14.

Sección II - JavaScript

JSON

1. De esta manera podemos convertir un texto JSON en un objeto de JavaScript

- [ ] `var obj = Object.parse(json);`
- [x] `var obj = JSON.parse(json);`
- [ ] `var obj = Object.Map(json);`
- [ ] `var obj = JSON.stringify(json);`

Funciones

2. ¿Qué obtenemos al ejecutar el siguiente código?

```
(function  f(f) {
return  typeof  f();
})(function(){ return  1; })
```

- **“function”**
- **“undefined”**
- **“number”**
- **Error**

¿Qué es lo que hace?

3. En JavaScript ¿qué hace la siguiente linea de código?

```
var square = number => number * number;
```

- **Nada, hay error de sintaxis**
- **Define una función sin parámetros que retorna el cuadrado de un número**
- **Define una función con dos parámetros que retorna el cuadrado de un número**

- [x] **Define una función que retorna el cuadrado de un número dado**

Verdades y mentiras

4. ¿Cuál de los siguientes enunciados **no es cierto** acerca de Javascript?

- [ ] **JavaScript es sencible a mayúsculas y minusculas**
- [ ] **JavaScript es un lenguaje debilmente tipado**
- [ ] **“do” es una palabra reservada del lenguaje**
- [x] **Al final de una sentencia en JavaScript es obligatorio usar “;”**

¿Qué imprime?

5. ¿Qué imprime en consola la siguiente porción de código?

```
function  f(x,  y=2,  z=7) {
return  x  +  y  +  z;
};
console.log(f(1) ===  10);
```

- [ ] **undefined**
- [ ] **NaN**

- [x] **true**
- [ ] **10**

6. De esta manera podemos convertir un objeto de JavaScript en un texto JSON

- [ ] `var json = Object.parse(obj);`
- [ ] `var json = JSON.parse(obj);`
- [ ] `var json = Object.Map(obj);`
- [x] `var json = JSON.stringify(obj);`

Ordenando

7. Al ejecutar las siguientes lineas de código ¿Qué resultado obtendremos?

```
let  letras  = ['Lambda',  'Alfa',  'Gamma',  'Beta'];
letras.sort();
```

- [x] **El arreglo letras se ordena de forma ascendente**
- [ ] **El arreglo letras se ordena de forma descendente**
- [ ] **Error, sort no es un método para los arreglos de JavaScript**
- [ ] **Error, el método sort requiere obligatoriamente una función que defina la manera**

Nombres

8. ¿Cuál es el resultado del siguiente código?

```
function  getName() {
const  name  =  'Juan';
name  =  'Jose';
return  name;
};
getName();
```

- [ ] **Juan**
- [ ] **Jose**
- [x] **error**
- [ ] **undefined**

9. Considera el siguiente código en JavaScript

```
function foo() {
return 5
}
```

¿Qué es lo que haria el codigo **let myVar = foo** ?

- [ ] **Asigna el valor entero 5 a la variable myVar.**
- [ ] **Lanza una excepción**
- [x]

Asigna una referencia hacia función foo en la variable myVar

- [ ] **Nada**

Null

10. En JavaScript ¿cuál es el tipo de dato de **null**?

- [ ] **undefined**
- [ ] **integer**
- [ ] **nill**
- [x] **object**

12. Resultado que imprime en consola

```
let  mascotas  = ["perro"  ,  "gato",  "loro",  "canario"]
;
mascotas.filter(mascota  =>  mascota.length >=  5);
```

- [x] `["perro", "canario"]`
- [ ] `["perro" , "gato", "loro", "canario"]`

13. ¿Cuando se ejecuta el siguiente código?

```
if(x  ===  y) {
console.log('Hola mundo');
}
```

- **Cuando las dos variables son iguales se imprime ‘Hola mundo’**
- **Cuando las variables y tipos son iguales se imprime ‘Hola mundo’**

14. ¿Cómo hacemos para agregar un `value` al final de un arreglo?

- **arr[arr.length + 1] = value;**
- **arr[arr.length] = value;**
- **arr[arr.length - 1] = value;**
- **arr = arr + value;**

15. ¿Cuál es un estado de promesa?

- **pendiente (pending)**
- **cumplida (fullfilled)**
- **rechazada (rejected)**
- **todas las anteriores**

16. A la posición 5 del arreglo de objetos llamado `persons` cambiar el valor de la propiedad lastName.

- **this.set(‘persons.4.lastName’, “juan”);**
- **this.set(‘persons.4.lastname’, “juan”);**
- **this.set(‘persons.4.last-name’, “juan”);**
- **this.set(‘persons[4]lastame’, “juan”);**

- **this.persons[4].lastname = “juan”;**

17. ¿Qué evento se usa para detectar cuando se pierde el focus en un input?

- **on-focus**
- **on-leave**
- **lost-focus**
- **on-blur**

18. ¿Qué seleccionas con esta linea de código?

```
document.querySelector("p.five")
```

- **El primer elemento** `p` con la clase `five`
- **Todos los elementos de la clase** `p`
- **Todos los elementos de** `p` con la clase `five`
- **Ninguno, este no es un selector valido**

19. ¿Qué método borra el elemento avion?

```
let trasportes = ['avion', 'camion', 'barco', 'tren'];
```

- `transportes.shift();`

> *El método shift() elimina el primer elemento del array y devuelve dicho elemento. Este método modifica la longitud del array.*

- `transportes.push('avion');`
- `transportes.pop();`
- `transportes.unshift();`

> *El método unshift() agrega uno o más elementos al inicio del array, y devuelve la nueva longitud del array.*

20. Cuando se asigna un observer a 2 propiedades ¿Qué tipo de observer es?

- **Simple**
- **Doble**
- **Complejo**
- **Binario**

21. A la posición 4 de un arreglo de objetos cambiar el valor de la propiedad lastname.

- **this.set('persons.4.lastname', "juan");**
- **this.set('persons.[4].lastname', "juan");**
- **this.set('persons[4]lastname', "juan");**
- **this.persons[4].lastname = "juan";**

22. ¿Cómo se declaran variables de bloque?

- **let**
- **var**
- **const**

23. ¿Cuál es el valor de `y` en el siguente bloque de codigo?

```
var x;
var y = x === null;
```

- [x] **false**

24. ¿Qué imprime en consola?

```
let x = "4" + 4 + 5;
let y = 4 + 4 + "5";
console.log(x + " " + y);
```

- [x] **445 85**

Sección III- Polymer

Dentro de un objeto

1. Para notificar a Polymer del cambio hecho solamente en una propiedad de un objeto debes invocar esta función

- [ ] **this.push**
- [ ] **this.notify**
- [x] **this.notifyPath**
- [ ] **this.set**
- [ ] **Ninguno de los anteriores**

Notify

2. Si en un componente tienes la siguiente propiedad

```
name: {
type: 'String',
value: '';
notify: true
}
```

¿Qué evento es lanzado al ejecutar la siguiente linea de código?

```
this.set('name', 'Juan Pérez');
```

- **name-modified**
- **name-notified**
- **name-changed**
- **Ningún evento es lanzado**

Propiedades en Polymer

3. Selecciona la opción que contenga sólo tipos de datos validos en Polymer

- **Boolean, Number, String, Array, Object**
- **Boolean, Number, String, Enum, Array**
- **Boolean, Computed, Number, String, Class**

- [ ] **Observer, Number, String, Array, Object**

Binding

4. ¿Qué tipo de “binding” es el siguiente?

```
<my-component attribute="[[property]]></my-component>"
```

- [ ] **Upward**
- [ ] **Two-way**
- [x] **One-way**
- [ ] **Lazy**

Estilizando

5. Si queremos que nuestro componente exponga la posibilidad de modificar su color de texto teniendo como valor por default el azul ¿cuál opción cubriria el requerimiento?

- [x] `:host { color: var(--my-text-color, blue) }`
- [ ] `:body { color: var(blue, --my-text-color) }`
- [ ] `:html { color: var(blue, --my-text-color) }`
- [ ] `:p, :label, :span { color: var(--my-text-color, blue) }`

Calculando

6. Son propiedades virtuales cuyo valor se determina basado en

otras propiedades

- **Observers**
- **Computadas**
- **Objects**
- **Behaviors**

Condiciones

7. Supongamos la siguiente condición para un requerimiento:
   Si la edad del usuario es mayor o igual a dieciocho el color de texto de su nombre debe ser azul, si no debe ser rojo.
   ¿cómo se podria imiplementar dicho requerimiento?

- **Usando un dom-if para cada una de las condiciones**
- **Usando un dom-if-else**
- **Usando un dom-if para una condición y un dom-else para la otra**
- **Todas las anteriores son válidas**

Removiendo

8. Supongamos que tienes un arreglo llamado persons: el cual tiene objetos como `{name: '', lastName: '' }`
   La longiutd del arreglo es 10. Tú necesitas remover el tercer elemento del arreglo

- [ ] `this.splice(persons, 2, 1);`
- [x] `this.splice('persons', 2, 1);`
- [ ] `this.[2] = null;`
- [ ] `Ninguna de las anteriores`

Filtrado

9. Permite el uso de una función **filter** como una de sus propiedades

- [ ] **dom-if**
- [x] **dom-repeat**
- [ ] **dom-module**
- [ ] **dom-filter**

Eventos

10. ¿Cuál es el nombre correcto del método para lanzar eventos en Polymer 2?

- [ ] **launchEvent**
- [ ] **fireEvent**
- [ ] **fire**
- [x] **dispatchEvent**

Notación

11. De forma declarativa ¿cómo haces el binding de la propiedad firstName?

- [ ] `<my-component firstName="Joe"></my-component>`
- [ ] `<my-component first.Name="Joe"></my-component>`
- [x] `<my-component first-name="Joe"></my-component>`
- [ ] `<my-component first-Name="Joe"></my-component>`

Observers

12. Los observers se ejecutan de forma:

- [ ] **Asincrona**
- [x] **Sincrona**
- [ ] **Sincrona y Asincrona**
- [ ]

  Los observers son valores de propiedades, por lo tanto no se ejecutan

Observando

13. Si estamos usando un **observer** para monitorear dos propiedades, estamos hablando de un **observer** tipo:

- [ ] **Simple**
- [ ] **Doble**
- [x] **Complejo**

- **Binario**

Cambios en propiedades

14. ¿Con que medios Polymer monitorea cambios en las propiedades…?

- **Return, New Properties, Data Bindings**
- **Observers, Molecule Elements, Template Repeater**
- **Observers, Computed Properties, Data Bindings**
- **Interfaces, Clases, Prototypes**

Focus

15. En polymer ¿cómo puedes añadir un listener a un campo de texto para detectar cuando se pierde el focus en un input?

- `<input type="text` on-blur=“myListener”>
- `<input type="text` on-focus=“myListener”>
- `<input type="text` on-leave=“myListener”>
- `<input type="text` lost-focus=“myListener”>

Otros

16. Cuál de los siguientes métodos no forma parte del ciclo de vidad de polymer?

- `constructor()`

- `connectedCallaback()`
- `disconnectedCallback()`
- `connectedEvent()`
- `attributeChangedCallaback()`