1. Parte 1: Orientación a Objetos (20 minutos)  
     
   1. Pregunta 1: Define brevemente los siguientes conceptos en el contexto de la programación orientada a objetos:  
      a. Clase 🡪 Describe los objetos.  
      b. Objeto 🡪 Información de algún dato de una clase.  
      c. Herencia 🡪 Mediante la palabra extends, hereda los objetos de una clase a otra.  
      d. Polimorfismo 🡪  
      e. Encapsulamiento 🡪   
     
   2. Pregunta 2: Escribe un ejemplo de una clase en Java que represente un "Coche" con al menos tres atributos y dos métodos.
2. public class Coche {
3. String Modelo;
4. String Color;
5. Boolean Nuevo;
6. public void arrancar() {};
7. public void cambiarColor() {};
8. }

Parte 2: Patrón Modelo-Vista-Controlador (MVC) (20 minutos)  
  
3. Pregunta 3:Explica en tus propias palabras qué es el patrón MVC y por qué es importante en el desarrollo de aplicaciones.

Se utiliza este patrón en la creación de aplicaciones ya que se divide la aplicación en partes y es más fácil para su creación. Es importante ya que si tienes algún fallo en alguna parte solo afecta a esa parte y no a las demás.  
  
4. Pregunta 4: Enumera y describe brevemente las responsabilidades de cada una de las partes del patrón MVC: Modelo, Vista y Controlador.

Modelo 🡪. Es donde se almacenan los datos.

Vista 🡪 Donde se realiza la interfaz para el usuario.

Controlador 🡪 Se encarga de conectar la vista con el modelo  
  
5. Pregunta 5: Proporciona un ejemplo de una situación en la que sería beneficioso utilizar el patrón MVC en el desarrollo de una aplicación.

Gestión de bancos.

Parte 3: Programación Asíncrona (20 minutos)  
  
6. Pregunta 6: Explica qué significa programación asíncrona y por qué es útil en el desarrollo de aplicaciones web.

La Programación Asíncrona es aquella que permite ejecutar varias tareas a la vez. Es útil ya que puedes realizar tareas mientras estas ejecutando otras.  
  
7. Pregunta 7: En JavaScript, ¿qué es una promesa (Promise)? ¿Cuál es su propósito principal y cómo se utiliza?  
  
8. Pregunta 8: Escribe un código JavaScript que demuestre cómo realizar una solicitud AJAX utilizando el método `fetch()` para obtener datos de una API externa y mostrarlos en una página web.  
  
  
  
Pregunta 9: Imagina que estás desarrollando una aplicación de lista de tareas pendientes. Debes escribir un algoritmo en Java o JavaScript que permita al usuario agregar nuevas tareas a la lista y marcar las tareas como completadas. El algoritmo debe incluir lo siguiente:

**Lo voy a realizar en JavaScript**  
  
a. Una estructura de datos para almacenar la lista de tareas.   
b. Funciones para agregar una nueva tarea a la lista.   
c. Funciones para marcar una tarea como completada.   
d. Un ejemplo de cómo se utilizaría este algoritmo en un programa principal.  
  
Puedes elegir implementar este algoritmo en Java o JavaScript, pero asegúrate de que esté claro y bien comentado.

<!DOCTYPE html>

<html lang="es">

<head>

    <meta charset="UTF-8">

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

    <title>Lista de Tareas</title>

    <script>

        // Estructura de datos para almacenar la lista de tareas

        let listaDeTareas = [];

        // Función para agregar una nueva tarea a la lista

        function agregarTarea() {

            let nuevaTarea = document.getElementById("nuevaTarea").value;

            if (nuevaTarea !== "") {

                listaDeTareas.push({ tarea: nuevaTarea, completada: false });

                mostrarTareas();

                document.getElementById("nuevaTarea").value = "";

            }

        }

        // Función para marcar una tarea como completada

        function marcarComoCompletada(indice) {

            if (indice >= 0 && indice < listaDeTareas.length) {

                listaDeTareas[indice].completada = true;

                mostrarTareas();

            } else {

                alert("El índice proporcionado no es válido.");

            }

        }

        // Función para mostrar las tareas en la lista

        function mostrarTareas() {

            let listaTareasElement = document.getElementById("listaTareas");

            listaTareasElement.innerHTML = "";

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

                let tarea = listaDeTareas[i];

                let tareaElement = document.createElement("li");

                tareaElement.textContent = tarea.tarea;

                if (tarea.completada) {

                    tareaElement.style.textDecoration = "line-through";

                }

                tareaElement.addEventListener("click", function() {

                    marcarComoCompletada(i);

                });

                listaTareasElement.appendChild(tareaElement);

            }

        }

    </script>

</head>

<body>

    <h1>Lista de Tareas</h1>

    <!-- Input para ingresar nueva tarea -->

    <input type="text" id="nuevaTarea">

    <!-- Botón para agregar tarea -->

    <button onclick="agregarTarea()">Agregar Tarea</button>

    <!-- Lista para mostrar las tareas -->

    <ul id="listaTareas"></ul>

</body>

</html>