

# Trabajo Práctico 2 - Alghoot Entrega 2

[7507/9502] Algoritmos y Programación III  
Curso 1  
Primer cuatrimestre de 2020

Kovnat, Leoni, Locatelli, Rosenblatt y Venglar.

## Índice

<b>1. Introducción</b>	<b>2</b>
<b>2. Supuestos</b>	<b>2</b>
2.1. Puntaje Negativo . . . . .	2
<b>3. Diagramas de Clases</b>	<b>2</b>
<b>4. Diagramas de Secuencia</b>	<b>6</b>
<b>5. Diagrama de Paquetes</b>	<b>8</b>
<b>6. Diagramas de Estados</b>	<b>8</b>
<b>7. Detalles de implementación</b>	<b>8</b>
7.1. Como se contesta una pregunta . . . . .	8
<b>8. Documentación</b>	<b>9</b>
8.1. Jugador . . . . .	9
8.2. Pregunta . . . . .	9

## 1. Introducción

El presente informe reúne la documentación de la solución del segundo trabajo práctico de la materia Algoritmos y Programación III que consiste en implementar el juego de trivia Kahoot, denominado por nosotros como Algohoot, utilizando los conceptos del paradigma de la orientación a objetos vistos hasta ahora en el curso.

## 2. Supuestos

### 2.1. Puntaje Negativo

Un jugador admite puntaje negativo si responde mal una pregunta con penalidad y tiene puntaje nulo.

## 3. Diagramas de Clases

Dejamos a continuación el diagrama de clases que representa las relaciones establecidas hasta el momento:

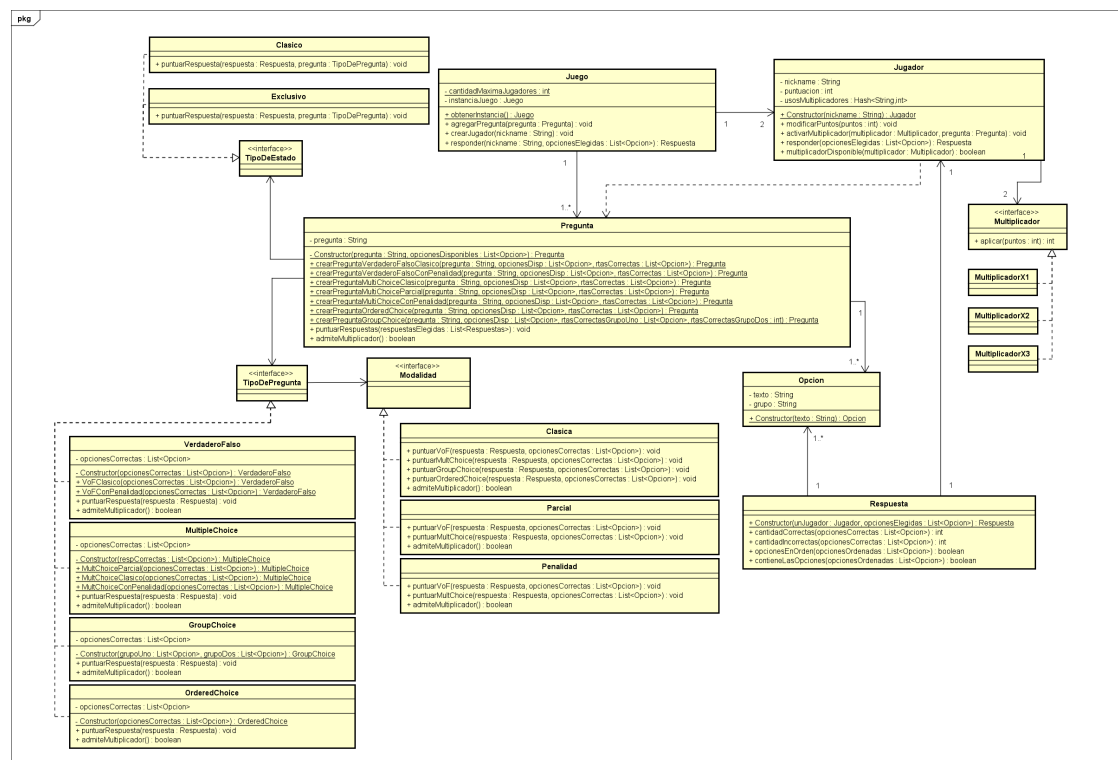


Figura 1: Diagrama de clases general.

Los siguientes diagramas muestran como las clases núcleo, es decir, las más importantes están relacionadas:

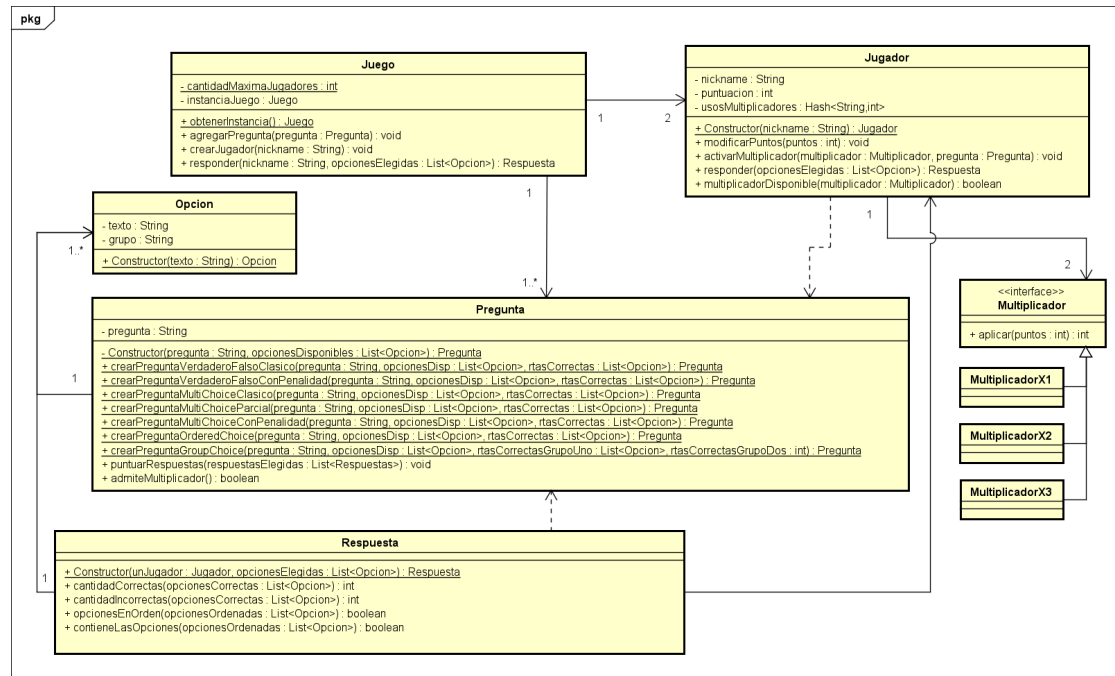


Figura 2: Principales relaciones entre Pregunta, Jugador, Respuesta, Opción, Multiplicador y Juego.

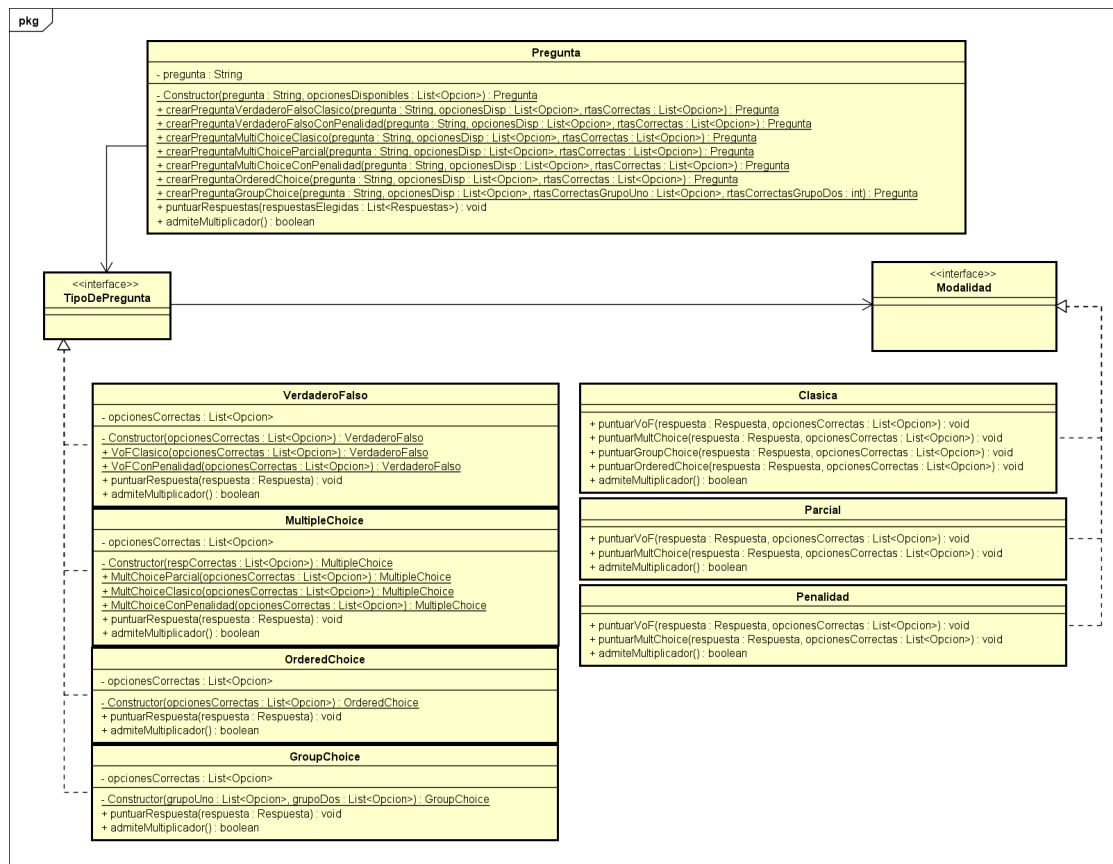


Figura 3: Cada instancia de Pregunta posee un tipoDePregunta, y esta última una instancia de la Modalidad.

Cada pregunta conoce el tipo de estado actual y este puede ser clásico o exclusivo:

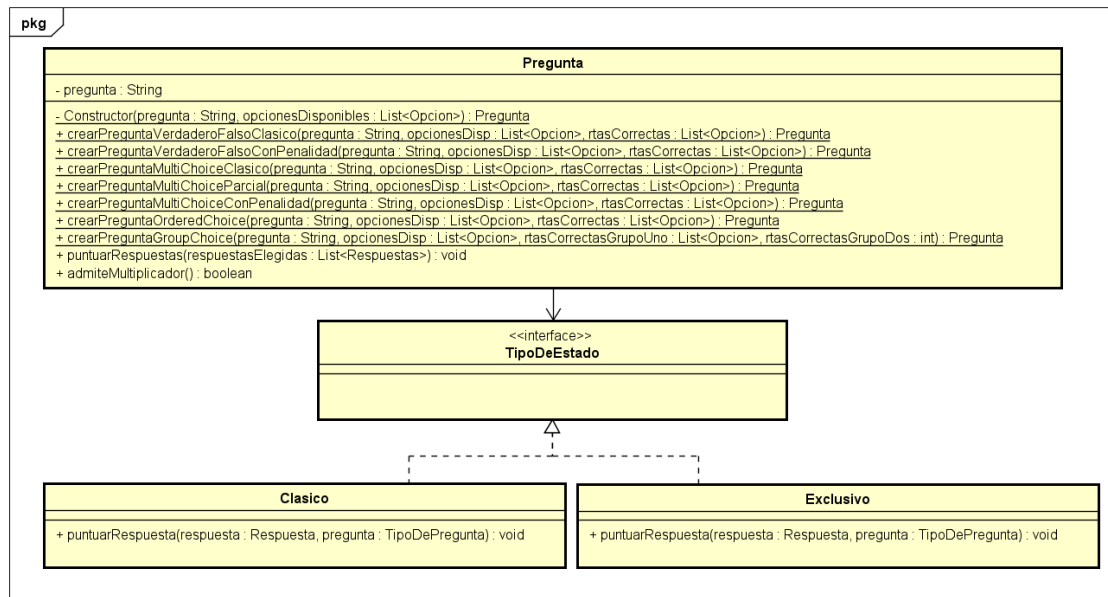


Figura 4: Diagrama de clases.

## 4. Diagramas de Secuencia

Dejamos a continuación los diagramas de secuencia que muestran las acciones más importantes:

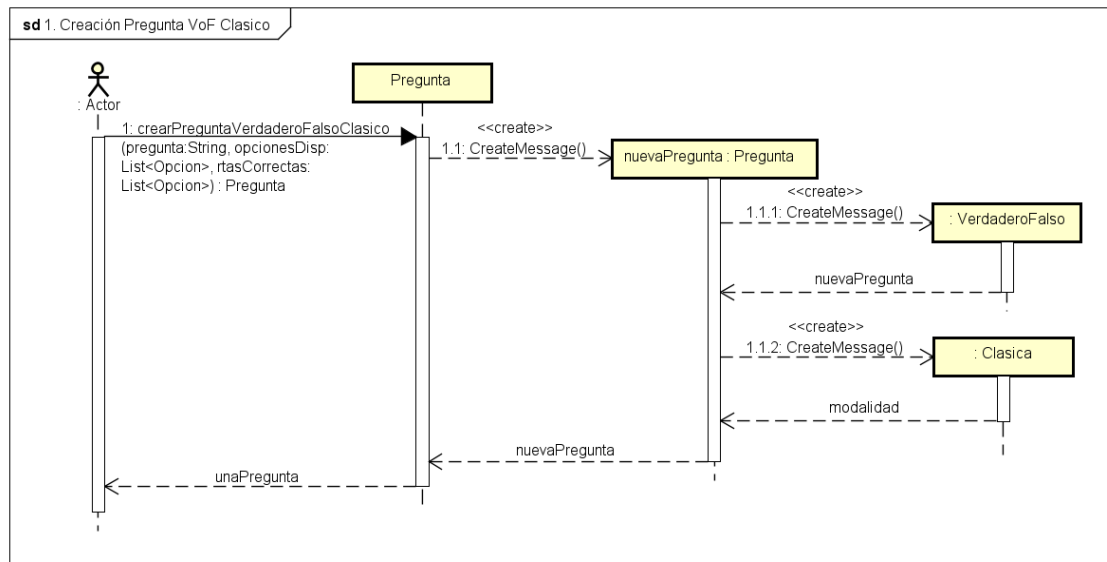


Figura 5: Creación de la instancia de Pregunta VoF Clasico.

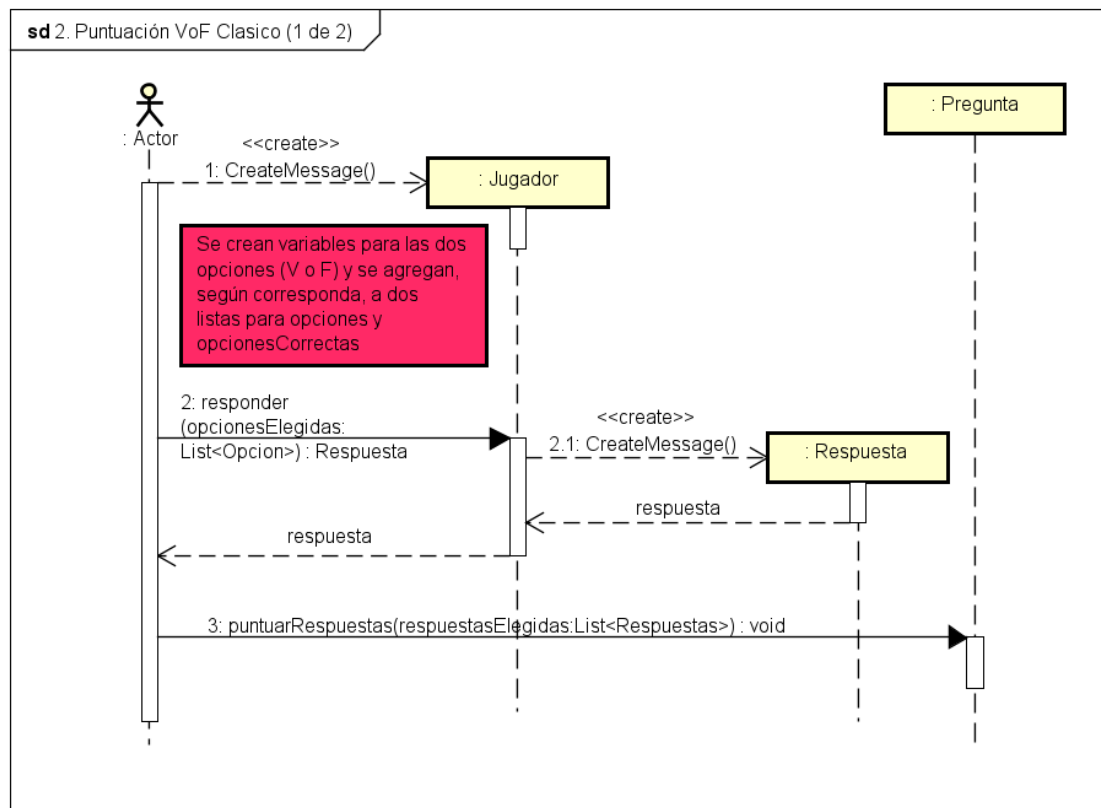


Figura 6: Evaluado de respuestas de Pregunta VoF Clasico (1/2).

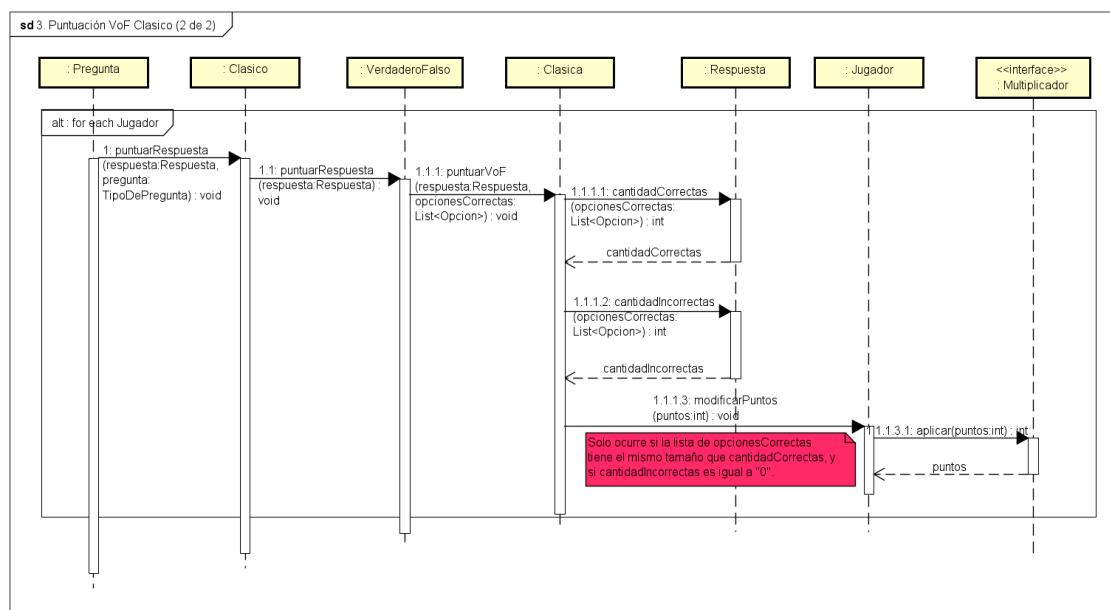


Figura 7: Evaluado de respuestas de Pregunta VoF Clasico (2/2).



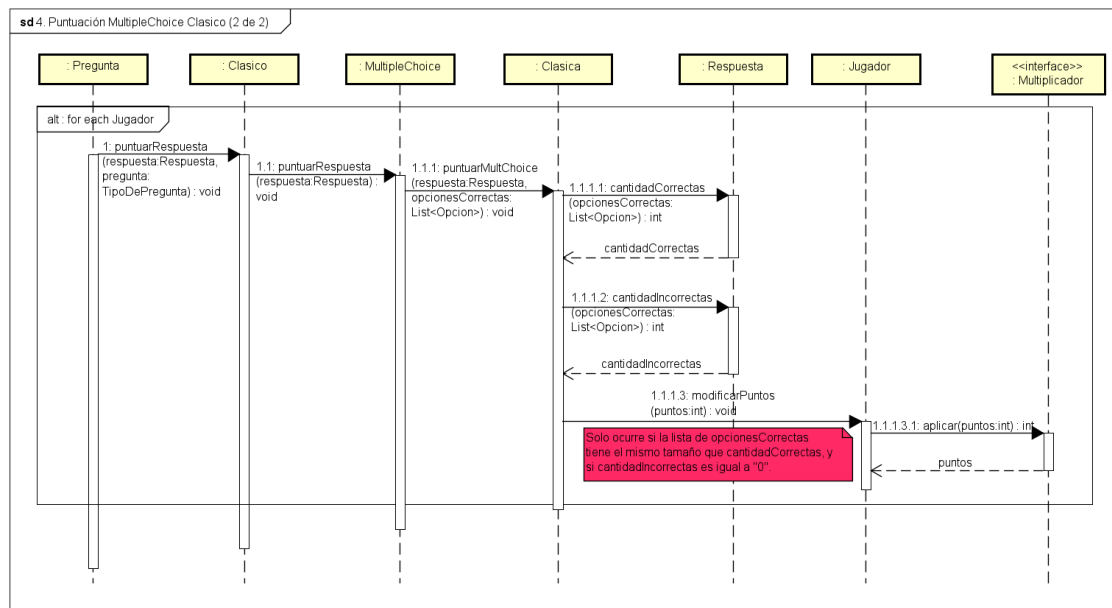


Figura 8: Evaluado de respuestas de Pregunta Multiple Choice Clasico.

## 5. Diagrama de Paquetes

No se solicita para esta entrega.

## 6. Diagramas de Estados

No se solicita para esta entrega.

## 7. Detalles de implementación

### 7.1. Como se contesta una pregunta

Una vez creada una Pregunta usando varios objetos Opcion, y creado uno o varios objetos Jugador, estos últimos generan cada uno un objeto Respuesta (mediante el método responder) el cual contiene una referencia al propio Jugador y una lista de (algunos o todos) los objetos Opcion usados en Pregunta. Esta lista corresponde a las opciones que el correspondiente Jugador eligió.

Después de eso se le da a Pregunta, mediante el método puntuarRespuesta, una lista compuesta de las Respuestas creadas por cada uno de los objetos Jugador. Entonces, puntuarRespuesta hace lo siguiente:

1. Itera por todas las Respuestas de la lista, y delega la verificación de las mismas, así como la correspondiente asignación de puntaje para cada Jugador, al objeto Estado (asociado a Pregunta), que a su vez se lo delega a TipoDePregunta, que a su vez se lo delega a Modalidad.
2. Por último, Modalidad se encarga de comparar los objetos Opcion de la pregunta con los objetos Opcion de la Respuesta, y asigna los puntos correspondientes al Jugador asociado a esa respuesta, usando el método puntuarRespuesta.

El hecho de que cada objeto `Pregunta` y `TipoDePregunta` tenga objetos asociados que definan el comportamiento del mismo (`Pregunta` tiene `Estado` y `TipoDePregunta`, mientras que `TipoDePregunta` tiene `Modalidad`), corresponde al patrón de diseño `Strategy`.

## 8. Documentación

### 8.1. Jugador

Cada instancia de `Jugador` tendrá un nombre, puntos y un multiplicador asociados al mismo. Se inicializa con puntaje en cero y nombre recibido por parámetro. Esta clase posee los métodos:

1. `public Respuesta responder(List<Opcion> opcionesElegidas);`

Recibe una lista de opciones seleccionadas por el jugador durante el turno vigente, y devuelve una instancia de `Respuesta` que posee una referencia al jugador y dichas opciones.

2. `public void modificarPuntos(int puntos);`

Este método recibe el número de puntos que obtuvo el jugador (ya sean positivos o negativos), se les aplica un multiplicador si el `Jugador` tiene uno activado, y se los añade.

### 8.2. Pregunta

Esta clase está formada por el texto literal de la pregunta, la lista de opciones disponibles, la lista con las opciones correctas su tipo y su modalidad.

Las preguntas se clasificarán por:

Tipo: "Verdadero o Falso" y "Multiple Choice"(hasta el momento).

Modalidad: Clásica, con puntaje Parcial y con Penalidad.

Por cada turno, la pregunta en curso recibirá una lista de respuestas emitidas por cada jugador.

1. `public static Pregunta crearPreguntaTipoYModalidad`

`(String pregunta, List<Opcion> opcionesDisponibles, List<Opcion> respuestasCorrectas);`

Recibe el texto de la pregunta, una lista de opciones disponibles y una lista de opciones correctas, y devuelve una instancia de `Pregunta` del tipo y modalidad especificados.

2. `public void puntuarRespuesta(List<Respuesta> respuestasElegidas);`

Este método recibe una lista con las respuestas elegidas por cada uno de los jugadores, evalúa la veracidad de cada una, y modifica el puntaje de cada jugador según corresponda.