**Diseño e implementación: Grupo 4**

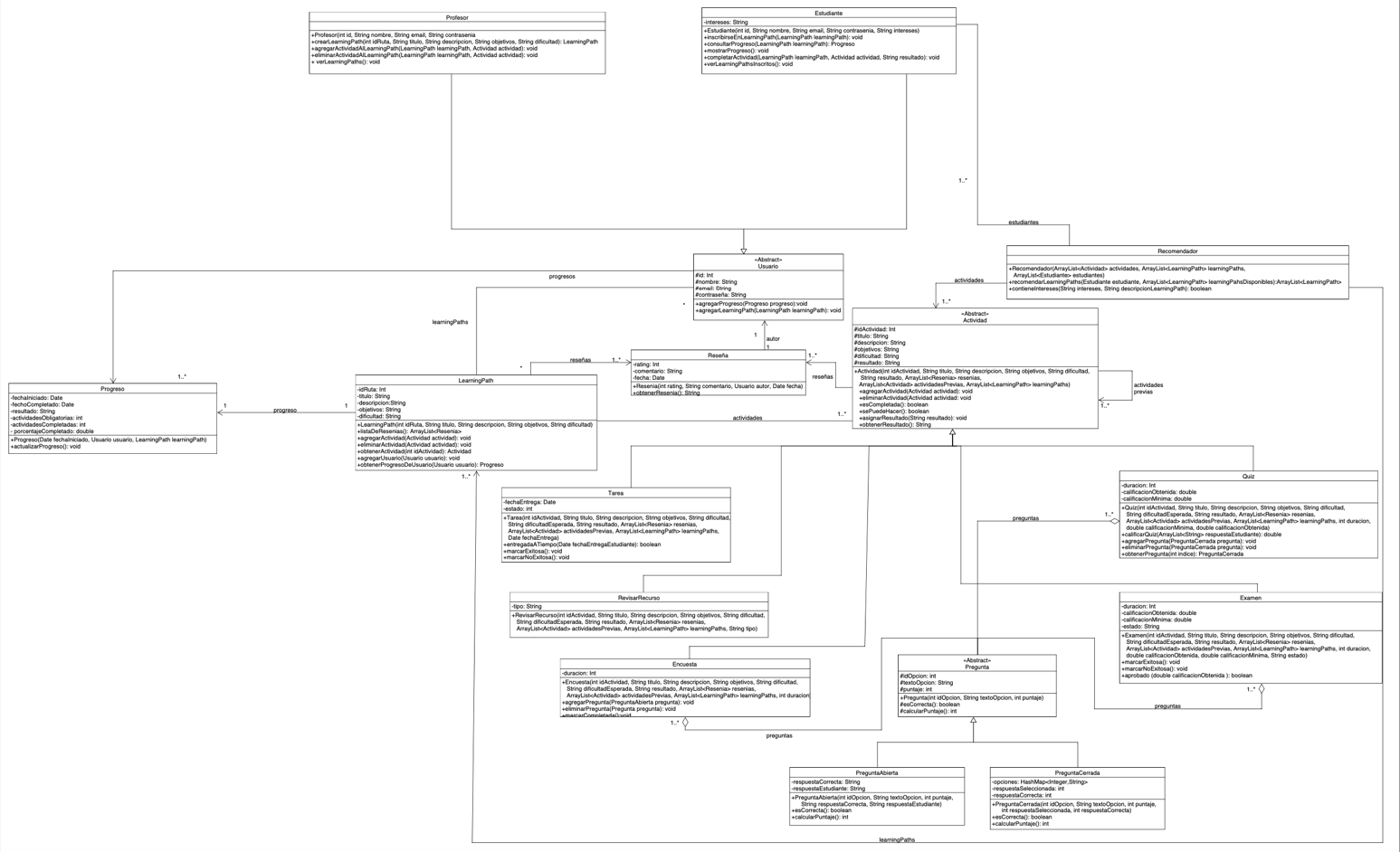
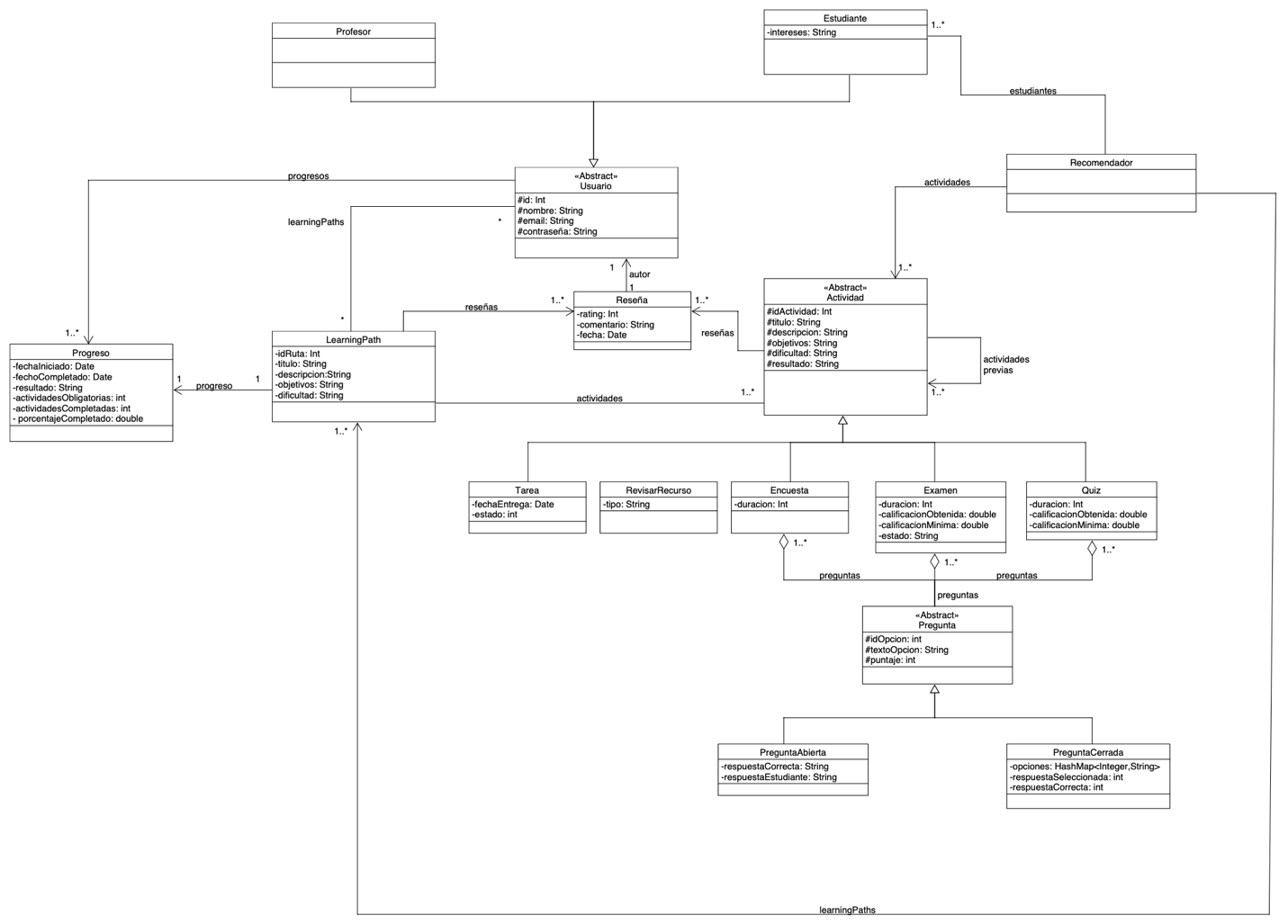
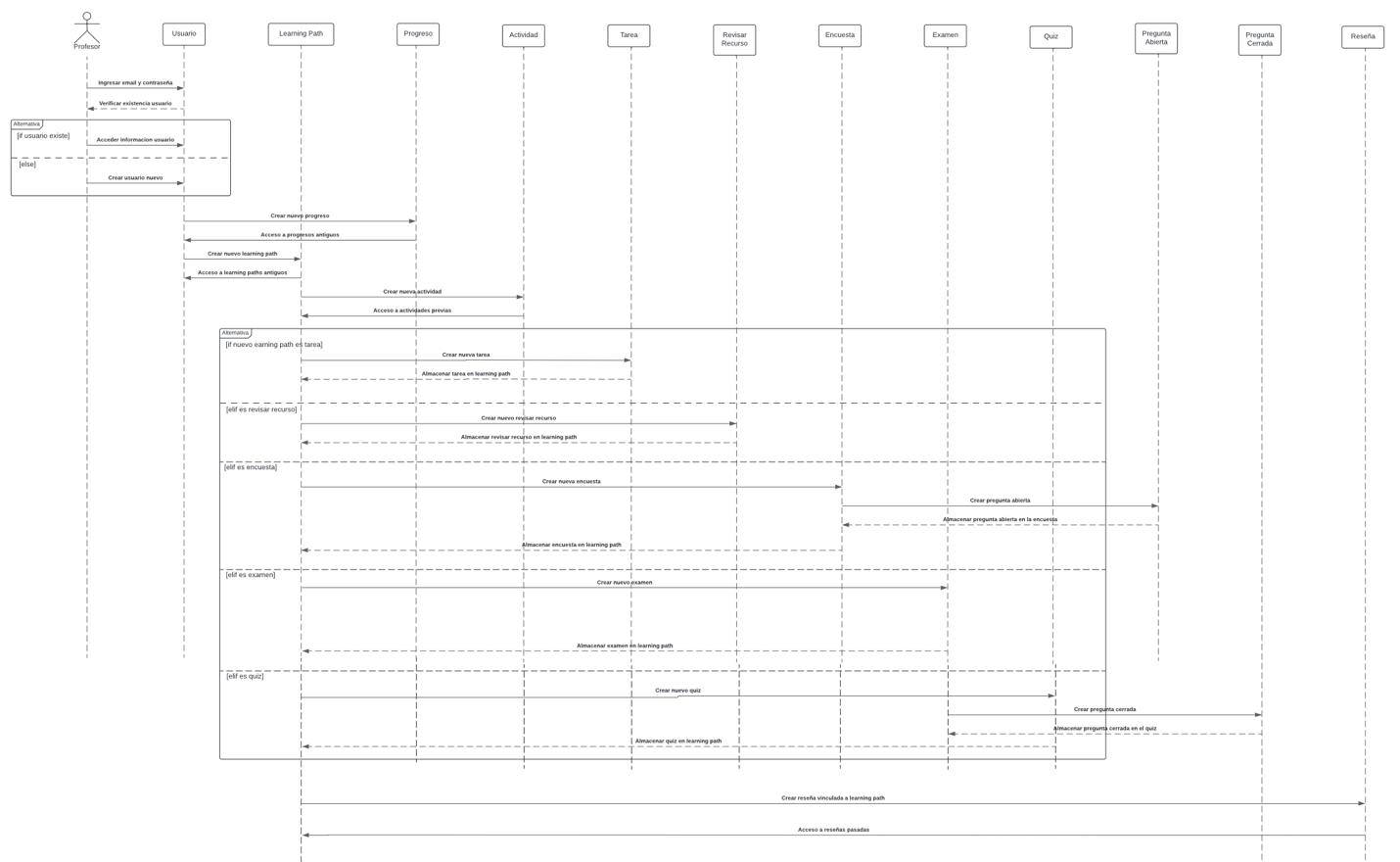
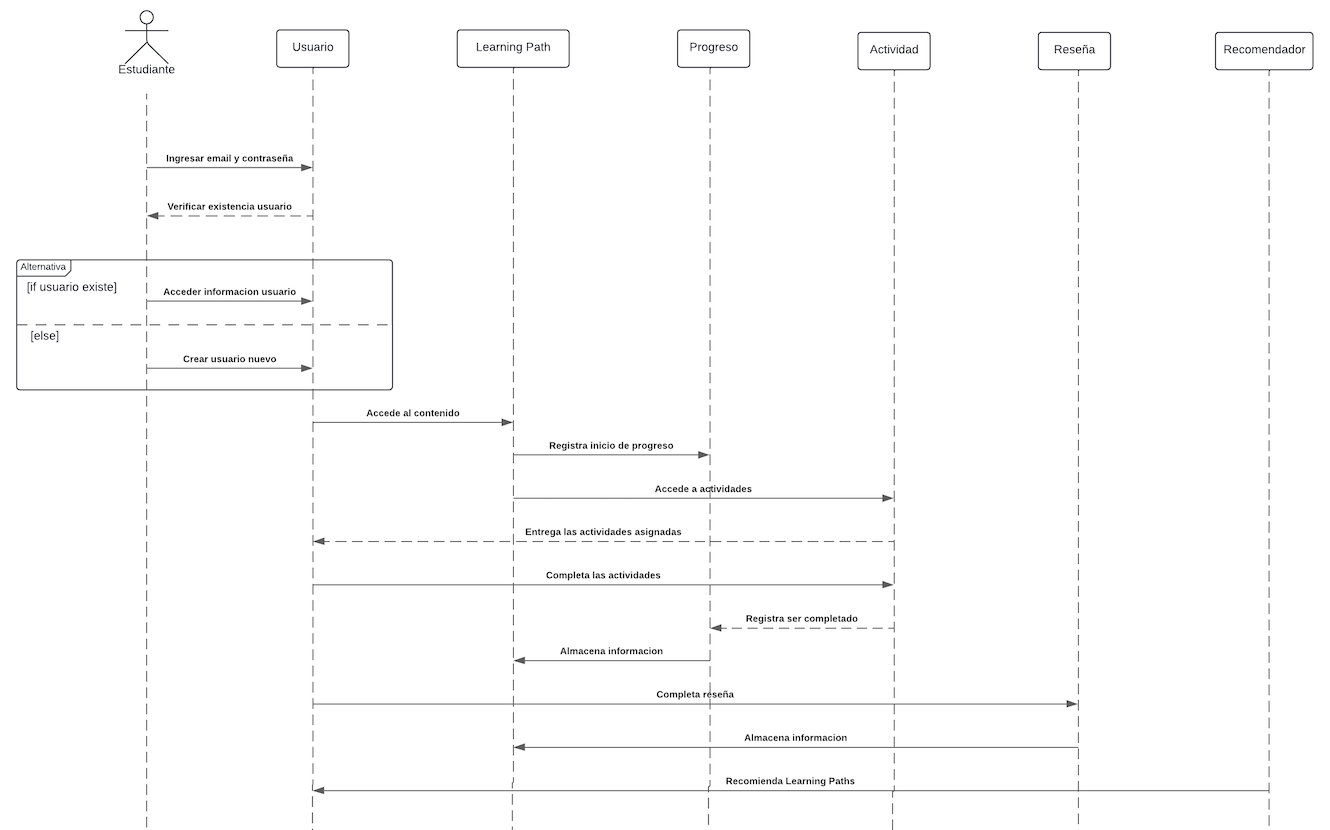
Diagrama de clases de diseño

Diagrama de clases de alto nivel

Diagramas de secuencia

Profesor:

Estudiante:



Justificación por clases

Usuario:

Es una clase que se definió como abstracta debido a que profesor y estudiante comparten atributos y métodos entre ellos, así mismo, comparten las mismas relaciones las cuales son el acceso a los progresos y a los learning paths (y por consiguiente a las actividades pero esto será nombrado después).

Métodos:

* agregarProgreso(…): Genera una nueva instancia para el registro de un progreso, registrando los datos necesarios de la clase Progreso.
* agregarLearningPath(…): Genera una nueva instancia para el registro de un Learning Path, registrando los datos necesarios de la clase LearningPath.

Profesor:

Esta clase solo difiere de estudiante en cuanto a los métodos y los accesos que posee sobre sus relaciones.

Métodos:

* Profesor(…): Registra un nuevo usuario profesor.
* crearLearningPath(…): Genera un nuevo Learning Path con la informacion requerida, registrada por el usuario profesor.
* agregarActividadAlLearningPath(…): Agrega la instancia de actividad nueva creada al Learning Path relacionado al profesor.
* eliminarlActividadAlLearningPath(…): Elimina la instancia de actividad referenciada en el LearningPath relacionado al profesor.
* verLearningPaths(…): Muestra los Learning Paths vinculados al usuario professor.

Estudiante:

Esta clase tiene un atributo propio llamado ‘intereses’ para que así se le puedan recomendar learning paths por medio de la clase Recomendador.

Métodos:

* Estudiante(…): Registra un nuevo usuario estudiante.
* inscribirseEnLearningPath(…): Inscribe al usuario estudiante al Learning Path de elección.
* consultarProgreso(…): Muestra el progreso en el Learning Path relacionado al usuario estudiante.
* mostrarProgreso(…): Muestra el progreso en el Learning Path relacionado al usuario estudiante.
* completarActividad(…): Genera acceso a la actividad que el usuario estudiante desea completar dentro de un Learning Path definido.
* verLearningPathsInscritos(…): Muestra todos los Learning Paths relacionados con el usuario estudiante.

Recomendador:

Esta es una clase que tiene relaciones con LearningPath, con Actividad y con Estudiante, se diseñó esta clase para que lea los intereses de cada estudiante y pueda filtrar una serie de recomendaciones para este usuario basado en un análisis de la descripción de learning paths y también de las actividades.

Métodos:

* Recomendador(…): Genera una nueva instancia de un estudiante recomendador.
* recomendarLearningPaths(…): Muestra las recomendaciones de Learning Paths sugeridas al estudiante de acuerdo a los progresos e intereses.
* contieneIntereses(…): Booleano que indica si el Learning Path contiene los intereses del estudiante.

Progreso:

Esta clase cumple la función de analizar el learning path al cual esta inscrito un estudiante y basado en las actividades que ha completado y las que le faltan puede dar un porcentaje de progreso, así mismo, también es capaz de guardar en que momento fue completado el learning path.

Métodos:

* Progreso(…): Genera una nueva instancia de progreso.
* actualizarProgreso(…): Actualiza la información del progreso.

LearningPath:

Puede considerarse una de las clases principales del proyecto, tiene relaciones a Reseña, Actividad, Progreso y Usuario, esta clase contiene un grupo de actividades con temáticas en común, también puede recibir reseñas tanto por profesores como por estudiantes y de ahí es que la relación sea a Usuario.

Métodos:

* LearningPath(…): Crea un nuevo Learning Path con toda la información necesaria ingresada por el usuario.
* ListaDeResenias(…): Muestra todas las reseñas existentes.
* agregarActividad(…): Genera una nueva instancia para una actividad.
* eliminarActividad(…): Elimina la actividad seleccionada.
* obtenerActividad(…): Obtiene la información relevante a la actividad en búsqueda.
* agregarUsuario(…): Crea un nuevo vinculo de acceso de un usuario a este Learning Path.
* obtenerProgresoDeUsuario(…): De acuerdo con la búsqueda ingresada, se obtiene el progreso relacionado a ese usuario.

Reseña:

Esta clase almacena una reseña que tiene rating, comentario un autor y un id, y una reseña puede corresponder tanto a un learning path como a una actividad.

Métodos:

* Resenia(…): Crea una nueva reseña con la información otorgada por el usuario.
* obtenerResenia(…): Obtiene la información relevante a la reseña en búsqueda.

Actividad:

Otra de las clases principales de este programa, tiene relación a Reseña, Learning path y a si misma como actividades existentes y actividades previas. También es una clase abstracta de la cual heredan otras 5 clases que serán mencionadas a continuación. Cabe aclarar que las actividades precias a una actividad son una sugerencia, un estudiante puede realizar una actividad sin haber realizado las previas antes, solo saldrá una advertencia.

Métodos:

* Actividad(…): Crea una nueva actividad con toda la información necesaria ingresada por el usuario.
* agregarActividad(…): Genera una nueva instancia para una actividad.
* eliminarActividad(…): Elimina la actividad seleccionada.
* esCompletada(): Booleano que indica si la pregunta fue completada.
* sePuedeHacer(): Booleano que indica si la pregunta está disponible para que el usuario estudiante pueda completar.
* asignarResultado(…): Asigna un resultado en escrito por el profesor de lo actividad.
* obtenerResultado(…): Se refleja el resultado previo.

Tarea:

Una de las clases que heredan de Actividad, la cual posee una fecha de entrega y un estado el cual puede ser ‘Exitosa’ o ‘No exitosa’.

Métodos:

* Tarea(…): Crea una nueva tarea con toda la información necesaria ingresada por el usuario.
* entregadaATiempo(…): Booleano que indica si fue entregada la tarea a tiempo o no.
* marcarExitosa(): Booleano que indica si la pregunta fue aprobada.
* marcarNoExitosa(): Booleano que indica que la pregunta no fue aprobada.

RevisarRecurso:

Una de las clases que heredan de Actividad, la cual tiene un tipo, que puede ser cualquier cosa como un video, pdf, etc. Esta clase es sencilla y solo se puede marcar como ‘Revisado’.

Métodos:

* revisarRecursos(…):

Encuesta:

Una de las clases que heredan de Actividad, que tiene una duración y una serie de preguntas de tipo PreguntaAbierta, que es una clase de la cual se hablara mas adelante. Esta actividad se marcara como ‘Completada’ al terminarla.

Métodos:

* Encuesta(): Crea una nueva encuesta con toda la información necesaria ingresada por el usuario.
* agregarPregunta(): Genera una instancia para adicionar una nueva pregunta.
* eliminarPregunta():
* marcarCompletada(): Booleano que indica que la encuesta fue completada.

Quiz:

Una de las clases que heredan de Actividad, la cual tiene una calificación mínima, un puntaje, una calificación obtenida y una serie de preguntqas cerradas las cuales pueden ser de tipo verdadero o falso.

Métodos:

* Quiz(…): Crea un nuevo quiz con toda la información necesaria ingresada por el usuario.
* calificarQuiz(): Calcula el puntaje y retorna el entero correspondiente.
* agregarPregunta(): Genera una instancia para adicionar una nueva pregunta.
* eliminarPregunta(): Elimina la pregunta referenciada.
* obtenerPregunta(): Muestra la pregunta referenciada.

Examen:

Una de las clases que heredan de Actividad, muy similar en aspectos a la clase Quiz la única diferencia es que las preguntas sonde tipo abiertas.

Métodos:

* Examen(…): Crea un nuevo examen con toda la información necesaria ingresada por el usuario.
* marcarExitosa(): Booleano que indica si la pregunta fue aprobada.
* marcarNoExitosa(): Booleano que indica que la pregunta no fue aprobada.
* aprobado(): Booleano que indica si el examen fue aprobado o no.

Pregunta:

Una clase abstracta que se creo porque pregunta abierta y cerrada comparten varios atributos entre ellas, esta clase se encarga de la modificación de los textos, opciones y puntajes, así mismo tiene métodos abstractos para saber si una pregunta es correcta y para calcular los puntajes.

Métodos:

* Pregunta(…): Crea una nueva pregunta con la información ingresada por el usuario.
* esCorrecta(): Booleano que indica si la respuesta es correcta o no.
* calcularPuntaje(..): Calcula el puntaje y retorna el entero correspondiente.

PreguntaAbierta:

Esta clase hereda de Pregunta, lo especifico de esta clase es la forma en la que calcula el puntaje y la forma de saber cuando es correcta o no.

Métodos:

* PreguntaAbierta(…): Crea una nueva pregunta con la información ingresada por el usuario.
* esCorrecta(): Booleano que indica si la respuesta es correcta o no.
* calcularPuntaje(): Calcula el puntaje y retorna el entero correspondiente.

PreguntaVerdaderoFalso:

Esta pregunta solo contiene dos respuestas posibles true o false, y hereda de la clase Pregunta

Métodos:

* PreguntaVerdaderoFalso (…): Crea una nueva pregunta con la información ingresada por el usuario.
* esCorrecta(): Booleano que indica si la respuesta es correcta o no.
* calcularPuntaje(): Calcula el puntaje y retorna el entero correspondiente.