**Práctica**

**Especificación técnica del código**

Programación

Natalia Rebeca Lara Robles

Domingo Martínez Díaz

Índice de contenidos

[1. Menú 1](#_Toc160623348)

[2. Método corrección de errores 2](#_Toc160623349)

[3. Clase Pregunta 3](#_Toc160623350)

[4. Clase Juego 4](#_Toc160623351)

[Método jugar 5](#_Toc160623352)

[Método mostrarPuntuaciones 5](#_Toc160623353)

[Método nuevaPregunta 6](#_Toc160623354)

[Método datosFicticios 6](#_Toc160623355)

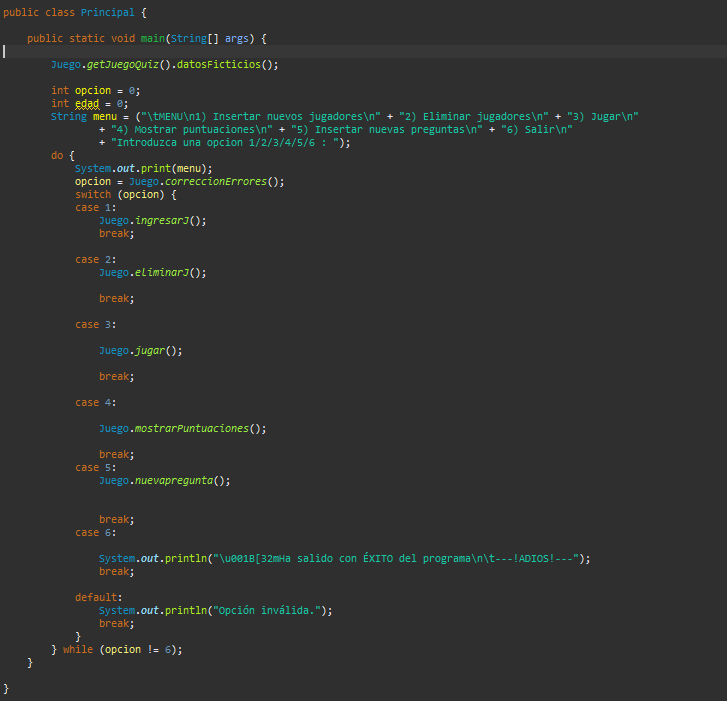
[Métodos ingresar y eliminar jugador 7](#_Toc160623356)

[5. Clase Jugador 8](#_Toc160623357)

[Método buscarJugador 9](#_Toc160623358)

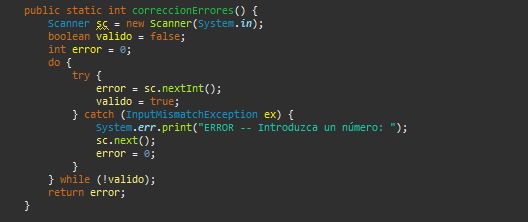
[Método toString 9](#_Toc160623359)

# Menú



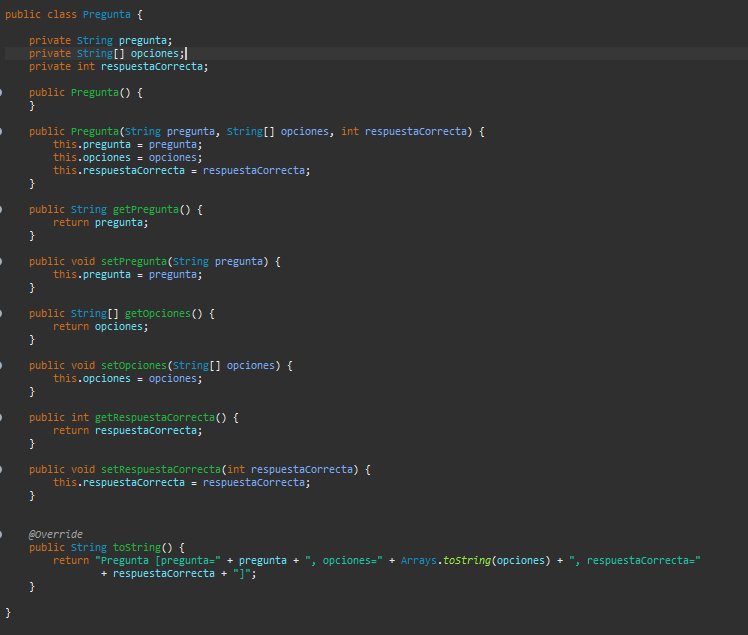
El programa tiene una clase principal llamada Principal que contiene el método main. En este método, se inicializan datos ficticios del juego y se muestra un menú al usuario. Se utiliza un bucle para permitir que el usuario elija entre varias opciones, como ingresar nuevos jugadores, eliminar jugadores, jugar, mostrar puntuaciones, agregar nuevas preguntas o salir del programa. Cada opción del menú llama a un método correspondiente en la clase Juego, que contiene la lógica para realizar estas acciones. Además, se manejan errores de entrada del usuario utilizando un método llamado correccionErrores(). El programa también utiliza códigos de escape ANSI para imprimir mensajes en color.

# Método corrección de errores



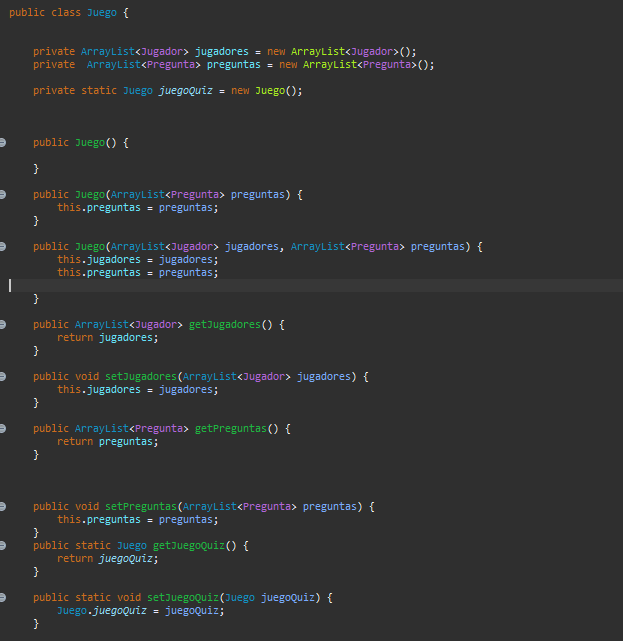
El método correccionErrores() de la clase Juego maneja errores en la entrada del usuario cuando se espera un número entero. Utiliza un bucle para solicitar la entrada del usuario continuamente hasta que se ingresa un número válido. Si se detecta un error, se muestra un mensaje de error y se solicita al usuario que ingrese nuevamente el número. Una vez que se ingresa un número válido, el método devuelve ese número.

# Clase Pregunta



La clase Pregunta es una representación de una pregunta dentro del contexto del juego de preguntas y respuestas. Contiene atributos para almacenar el enunciado de la pregunta, un conjunto de opciones de respuesta y la posición de la respuesta correcta dentro de este conjunto. Los métodos de acceso permiten obtener y establecer estos atributos de manera segura. Además, la clase incluye un método toString() que proporciona una representación legible de la pregunta, mostrando el enunciado, las opciones y la respuesta correcta. En resumen, la clase Pregunta encapsula la información necesaria para definir una pregunta en el juego, proporcionando funcionalidades para manipular y visualizar dicha información de manera coherente.

# Clase Juego



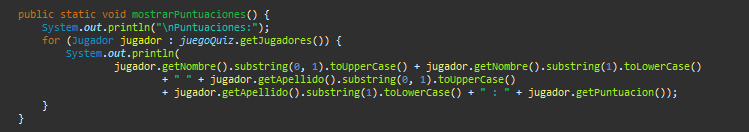
## La clase Jugador representa a un participante en el juego, almacenando información relevante como su nombre, apellido, edad y puntuación. Está diseñada para proporcionar métodos que permiten acceder y modificar estos datos de manera segura. Además, incluye un método estático buscarJugador() que facilita la búsqueda de un jugador en una lista dada su nombre y apellido. Esto proporciona flexibilidad al interactuar con una colección de jugadores en el juego. Por último, el método toString() permite obtener una representación legible del jugador, incluyendo su nombre, apellido, edad y puntuación.

## Método jugar



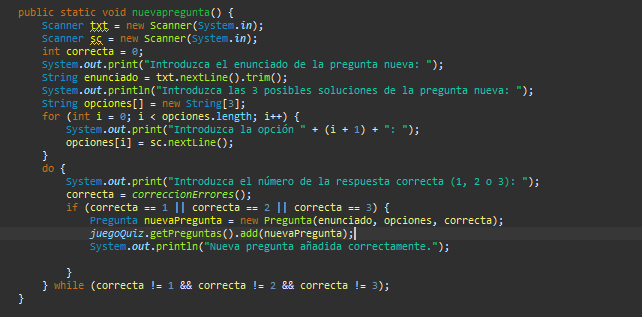
El método jugar es nuestra acción de jugar dentro del programa, este te pintara los jugadores por pantalla, elegimos uno escribiendo su nombre y apellidos y nos generara las preguntas aleatoriamente. Las preguntas se responden con 1/2/3 eligiendo cual sería la correcta. La respuesta correcta la tenemos guardada en un entero, esta se comprueba con la respuesta dada por el jugador y si es correcta añade 10pts al atributo puntuación del jugador seleccionado.

## Método mostrarPuntuaciones



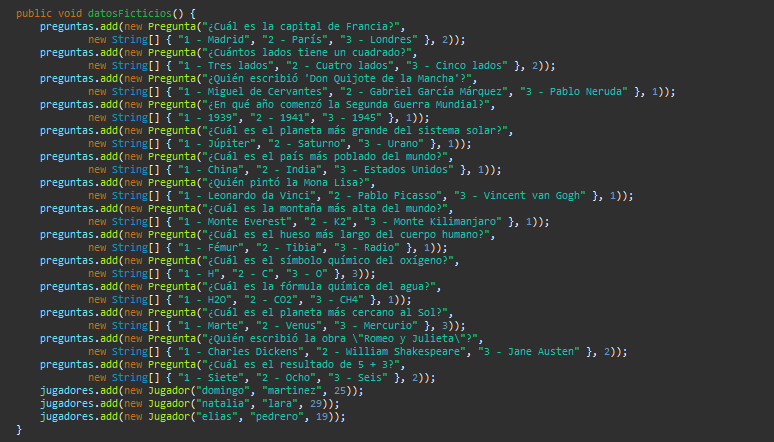
Método de tipo void dentro de la clase Juego donde se pintan las puntuaciones de los jugadores junto a su nombre, utilizamos substring para poner en mayúscula la primera letra de los nombres y que sea más legible para el usuario.

## Método nuevaPregunta



Método en la clase Juego para crear una nueva pregunta. Pediremos al usuario que nos diga una pregunta y las 3 opciones de respuesta. Ahora le pediremos que nos indique cuál de las 3 opciones de respuesta es la correcta, lo recogeremos con el método correccionErrores () . Todos estos datos se añadirán se le asignaran a un objeto de tipo Pregunta nuevaPregunta que debemos declarar para después añadirlo a nuestro array preguntas a través del método getPreguntas()

## Método datosFicticios



El método datosFicticios() en la clase Juego es utilizado para inicializar la lista de preguntas y jugadores con datos predefinidos. Este método se llama al inicio del juego llamando Juego.juegoQuiz para proporcionar una base inicial de preguntas y jugadores.

Dentro del método datosFicticios(), se añaden preguntas predefinidas. Cada pregunta se crea utilizando el constructor de la clase Pregunta y se agrega a la lista de preguntas del juego.

También se añaden jugadores predefinidos con nombres, apellidos y edades ficticias. Cada jugador se crea utilizando el constructor de la clase Jugador y se agrega a la lista de jugadores del juego

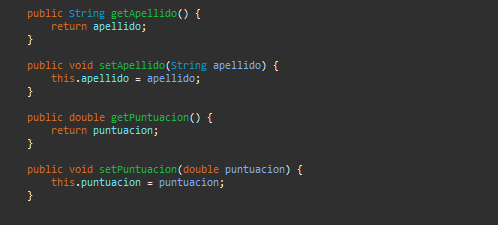
## Métodos ingresar y eliminar jugador



El método ingresarJ() de la clase Juego permite agregar un nuevo jugador al juego. Primero, solicita al usuario que ingrese el nombre, apellido y edad del nuevo jugador. Luego, utiliza el método estático correccionErrores() para asegurarse de que la entrada de la edad sea un número válido. Después de recibir y validar la información del nuevo jugador, crea un objeto Jugador con estos datos y lo agrega a la lista de jugadores del juego.

Por otro lado, el método eliminarJ() permite eliminar un jugador existente del juego. Comienza mostrando una lista de todos los jugadores disponibles. Luego, solicita al usuario que ingrese el nombre y apellido del jugador que desea eliminar. Utiliza el método estático buscarJugador() para buscar al jugador con el nombre y apellido especificados en la lista de jugadores. Si encuentra al jugador, lo elimina de la lista. En caso de que el jugador no exista en la lista, se muestra un mensaje indicando que el jugador no se encuentra en la lista de jugadores disponibles.

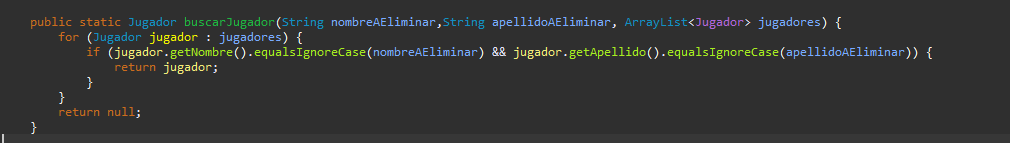
# Clase Jugador



La clase Jugador representa a un participante en el juego. Los atributos de la clase incluyen nombre, apellido, edad y puntuación, todos ellos almacenados en variables de instancia privadas.

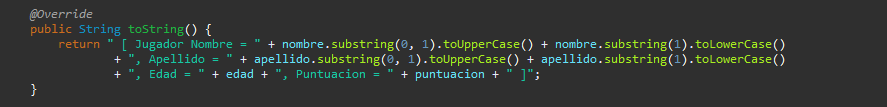
La clase proporciona varios constructores para inicializar un objeto Jugador con diferentes combinaciones de atributos. También incluye métodos de acceso (getters) y modificación (setters) para cada atributo, lo que permite obtener y establecer los valores de los atributos del jugador desde fuera de la clase.

## Método buscarJugador



La clase contiene un método estático llamado buscarJugador(), que toma el nombre y apellido de un jugador junto con una lista de jugadores y devuelve el jugador correspondiente si se encuentra en la lista, o null si no se encuentra.

## Método toString



El método toString() está editado para proporcionar una representación en forma de cadena del objeto jugador. Este método devuelve una cadena que muestra el nombre y apellido del jugador con la primera letra en mayúscula, seguida de su edad y puntuación.