Algoritmos y Programación I
Primer Examen Practico- 2018-2

Nombre:	
Código:	**Fecha: 28 de Septiembre de 2018**

Unidad 1: Introducción a la solución de problemas utilizando algoritmos

OE1.5. Utilizar expresiones simples: declaraciones de variables, asignaciones e invocaciones a métodos. Esto implica entender los conceptos de parámetro y de creación de objetos.

OE1.6. Utilizar un ambiente de desarrollo de programas (incluyendo la compilación y ejecución de programas desde consola) y un espacio de trabajo predefinido, para construir la solución de un problema.

OE1.7. Interpretar la documentación (modelos conceptuales y especificación de las operaciones) de una solución con el fin de: a) llevar a cabo la implementación con base en el diseño (rol del programador vs rol del diseñador), b) comprender el desarrollo en su totalidad, c) verificar la coherencia entre análisis, diseño e implementación (trazabilidad de los elementos en los artefactos del desarrollo) y, d) validar la solución propuesta en el diseño con las necesidades del usuario expuestas en los requerimientos funcionales.

OE1.8. Aplicar el concepto de encapsulamiento en el modelamiento e implementación de soluciones bajo el paradigma de programación orientado a objetos.

OE1.10. Declarar y utilizar variables diferenciando entre tipos de datos primitivos y tipos de datos referenciados.

OE1.11. Construir las clases que implementan el modelo de la solución del problema.

OE1.12. Construir las clases que implementan una interfaz de usuario (consola) e integrarlas con las clases que implementan el modelo de la solución del problema.

OE1.13. Utilizar una arquitectura para un programa que permita repartir de manera adecuada las responsabilidades entre la interfaz de usuario y el modelo de la solución. El estudiante deberá poder explicar la importancia de mantener separadas las clases de estos dos dominios.

Unidad 2: Solución de problemas utilizando estructuras de control condicionales

OE2.2. Utilizar expresiones aritméticas, lógicas, relacionales y operaciones con cadenas en el cuerpo de un método.

OE2.3. Declarar y hacer llamados a métodos constructores declarados explícitamente en la misma clase y utilizar null en caso de que un objeto no haya sido inicializado.

OE2.4. Utilizar las instrucciones condicionales como parte del cuerpo de un algoritmo, para poder considerar distintos casos de la solución de un problema.

OE2.6. Interpretar los errores producidos en tiempo de ejecución para el caso en que se realicen llamados u operaciones con objetos que no han sido construidos.

OE2.8. Valorar la coherencia estricta entre el diseño propuesto en el diagrama de clases y la implementación en el lenguaje de programación.

Enunciado

League of Legends (también conocido por sus siglas LoL) es un videojuego de género multiplayer online battle arena (MOBA) y deporte electrónico desarrollado por Riot Games para Microsoft Windows y OS X.

Los jugadores compiten en partidas, que duran entre 20 y 60 minutos en promedio. En cada modo de juego, los equipos trabajan juntos para lograr una condición de victoria, normalmente destruyendo la estructura central (llamado Nexo) en la base del equipo enemigo después de pasar por alto una línea de estructuras defensivas llamadas Torretas. En todos los modos de juego, los jugadores controlan personajes llamados «campeones», elegidos o asignados en cada partida, que tienen un conjunto de habilidades únicas, con los cuales jugarán toda la partida hasta su conclusión. Los campeones dentro del juego interactúan entre sí consiguiendo Asesinatos (kills), asistencias (Assists) y muertes (Deaths) las cuales serán acumuladas en el jugador que controla el campeón.

El equipo de Riot se ha comunicado contigo porque quieren implementar una aplicación que permita calcular algunas estadísticas sobre los jugadores, la primera de ellas es el KDA. El KDA se utiliza para medir el desempeño general de los jugadores en las partidas, es un promedio que toma en cuenta los Asesinatos, las Muertes y las Asistencias, de hecho su nombre viene de las siglas en inglés de estos [K]ill, [D]eath y [A]ssist. Este se calcula utilizando la siguiente fórmula:

KDA =(Kills + Assists) / Deaths

Como se puede ver simplemente suma los asesinatos con las asistencias y se divide entre las muertes. Ten en cuenta que si la **cantidad de muertes es 0**, se divide por 1 y no por 0, para no producir un error Aritmético. Se espera que al hacer los cálculos relacionados con el KDA se muestre en consola el nickname y valor del KDA del jugador con mejor KDA y el de peor KDA.

La segunda funcionalidad que se desea tener el programa es poder eliminar una de los jugadores del programa dado su nickname. El equipo de Riot logró adelantar la fase de análisis y de diseño, y te hace entrega de todos los documentos generados en ellas, tu labor es ahora hacer la etapa de implementación:

Requerimientos Funcionales

El programa debe estar en la capacidad de:

Nombre:	R1. Mostrar los jugadores con mejor y peor KDA
Resumen:	Permite mostrar en la consola los jugadores que tienen el mejor y peor KDA.
Entradas:	*ninguna*
Resultado:	Se muestra en consola

Nombre:	R2. Eliminar un jugador
Resumen:	Se elimina del sistema el jugador que tiene el nickname igual al ingresado por el usuario. En caso de que no exista un jugador con ese nickname se informa al usuario que no se pudo encontrar ese jugador.
Entradas:	El nickname del jugador a ser borrado: String
Resultado:	Un mensaje en consola que confirma que se eliminó el jugador en caso de que exista, o un mensaje que diga que no fue encontrado el jugador en caso contrario.

Diagrama de Clases

El diagrama de clases se presenta a continuación:

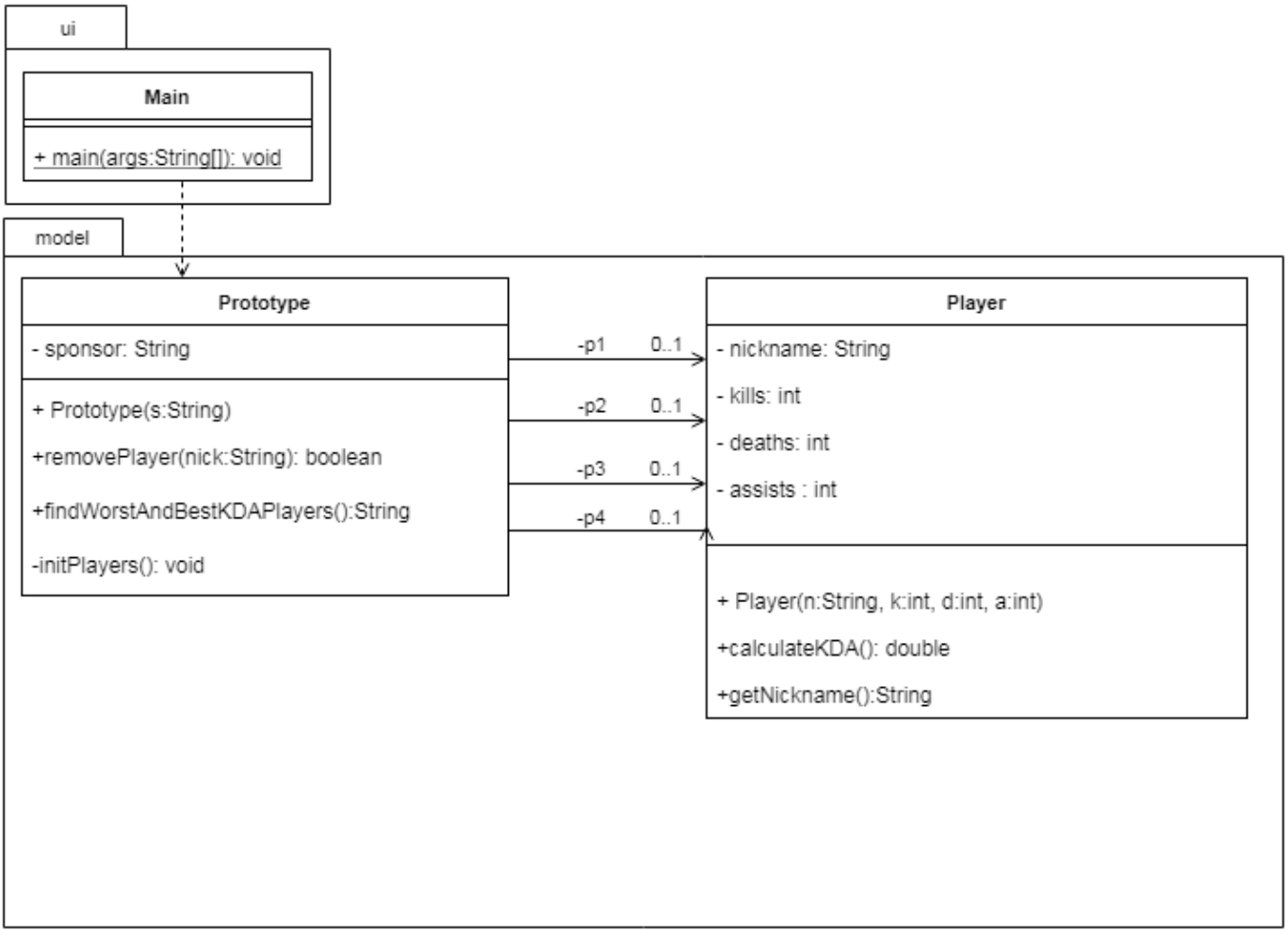
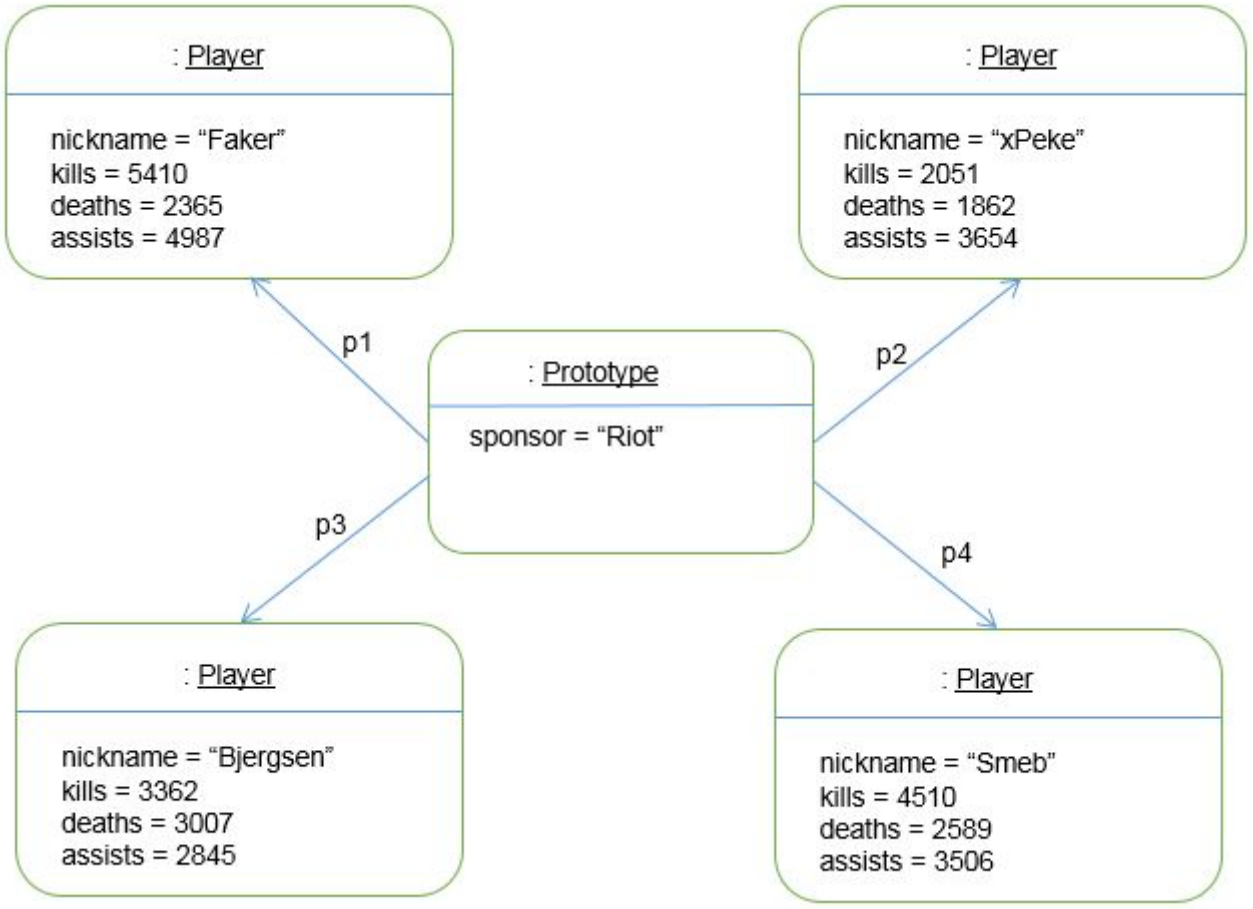


Diagrama de Objetos

Un diagrama de objetos con base en el diagrama de clases anterior se presenta a continuación, este describe el estado inicial del programa:



Entrega

Usted debe entregar un archivo en formato .zip cuyo nombre debe tener el formato PRIMERAPELLIDO_PRIMERNOMBRE.zip en el cual se encuentre una carpeta con el nombre API_PRACTICO1_PRIMERAPELLIDO_PRIMERNOMBRE, dentro de la cual se encuentren las carpetas de los respectivos paquetes definidos en el diagrama de clases y éstas a su vez contengan los archivos de las clases implementadas en Java.

En el método main usted debe implementar un escenario a través de la creación de los objetos con los valores predeterminados que considere necesarios y utilizando lo métodos presentados en el diagrama de clases. Una vez realizado ésto el programa deberá ejecutar el requerimiento 1, después el requerimiento 2 lo cual implica leer por consola los valores que considere necesarios y después terminar su programa.