Manager de miembros y rutinas de gimnasio

# Análisis del Sistema (30 puntos)

## 1.1 Requisitos funcionales del sistema (5 pts)

Escribir los puntos exactos que debe cumplir tu sistema.

Ejemplo: “El usuario puede crear una nueva solicitud de boletos”, etc.

- El usuario debe ser capaz de crear un nuevo miembro del gimnasio y agregarlo al sistema.

- El usuario debe ser capaz de ingresar un entrenador del gimnasio al sistema.

- El usuario debe ser capaz de ingresar un nuevo ejercicio al sistema.

- El usuario debe ser capaz de crear una nueva rutina de ejercicios, usando los ejercicios que ya se encuentran ingresados en el sistema.

- El usuario debe ser capaz de asignar un miembro del gimnasio a un entrenador, con una rutina seleccionada por el miembro. Esto no se puede hacer si el miembro del gimnasio ya tiene un entrenador asignado previamente.

- El usuario debe ser capaz de desasignar un miembro del gimnasio con un entrenador.

- El usuario debe ser capaz de saber que entrenador tiene más alumnos, que entrenador tiene menos alumnos, que rutina es la más usada, que rutina es la menos usada, y el porcentaje de entrenadores que se encuentra sobrecargado.

## 1.2 Clases necesarias y su propósito (5 pts)

|  |  |
| --- | --- |
| Clase | Propósito |
| Miembro | Esta clase se refiere a un miembro del gimnasio, el cual tiene rutina y entrenador. Sus características son el nombre, la edad, la cantidad de peso actual, la meta de peso que se tiene, el tipo de membresía que se compró, la rutina que esta usando y el nombre del entrenador que tiene asignado. Como métodos puede asignarse o desasignarse de un entrenador. |
| Entrenador | Esta clase se refiere a un entrenador del gimnasio. Sus características son el nombre, la edad, la cantidad de alumnos que tiene, las rutinas que el entrenador usa, la cantidad máxima de alumnos que el entrenador puede tener y si el entrenador está sobrecargado o no. Como métodos puede añadir o quitarse un alumno (que es un miembro del gimnasio), y añadir o quitarse una rutina. |
| Ejercicio | Esta clase contiene todas las características con las que cuenta un tipo de ejercicio. Estas características son su nombre, el tipo de ejercicio que es, el nivel de cansancio que genera, el número de repeticiones del ejercicio y la cantidad de calorías que ayuda a quemar. |
| Rutina | Esta clase representa una rutina de ejercicio, que puede contener varios Ejercicios. Sus características son su nombre y una lista con los ejercicios que incluye. Como métodos, puede añadirse o quitarse un ejercicio de la rutina, y consultar los ejercicios que la rutina usa. |
| Contador de ejercicios | Esta clase sirve como auxilar de Rutina, y su propósito es regitsrar cuantas veces tiene asignado una rutina un entrenador. Sus características son el nombre de la rutina de la que lleva un registro, y la cantidad de veces que se usa dicha rutina. |
| Controlador | Esta clase encarga de dirigir las demás clases. Se comunica con los modelos de Miembro, Entrenador, Ejercicio y Rutina, y tiene acceso a sus métodos. Esta clase cuenta con una lista de Miembros, Entrenadores, Ejercicios y Rutinas, y tiene diferentes métodos para obtener estadísticas propias del sistema. |
| Vista | Esta clase se encarga de recibir y mostrar información al usuario. Se comunica con el Controlador principal, el cual realiza las operaciones que el programa requiere. |

## 1.3 Atributos de cada clase (10 pts)

Clase: Miembro

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo de dato | Visibilidad | Propósito |
| Nombre | String | Privada | Es el nombre del miembro del gimnasio. Se usa como identificador |
| Edad | int | Privada | Es la edad del miembro del gimnasio |
| Peso actual | double | Privada | Es el peso actual del miembro de gimnasio. |
| Peso ideal | double | Privada | Es el peso al que el miembro del gimnasio quiere llegar |
| Tipo de membresía | String | Privada | Es el tipo de membresía con la que el usuario cuenta. Esta puede ser Basica, Premium o Master |
| Rutina de ejercicio | String | Privada | Es el nombre de la rutina de ejercicio que el miembro está usando actualmente. No se puede tener una rutina sin un entrenador |
| Entrenador | String | Privada | Es el nombre del entrenador con el que el usuario está asignado. |

Clase: Entrenador

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo de dato | Visibilidad | Propósito |
| Nombre | String | Privada | Es el nombre del entrenador. Funciona como identificador |
| Edad | int | Privada | Es la edad del entrenador |
| Lista de alumnos | ArrayList<String> | Privada | Es una lista que contiene el nombre de todos los alumnos que el entrenador tiene asignados. |
| Rutinas de ejercicio | ArrayList<Contador de ejercicios> | Privada | Es la lista de rutinas que el entrenador usa actualmente. Incluye únicamente el nombre, no tiene los datos de la rutina en sí. |
| Cantidad máxima de alumnos | int | Privada | Es la cantidad máxima de alumnos que un entrenador tiene. Es posible asignarle más alumnos de los que su máximo indica, pero esto lo volverá “sobrecargado” |
| Sobrecargado | boolean | Privada | Indica si el entrenador está sobrecargado “true” o no “false” |

Clase: Ejercicio

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo de dato | Visibilidad | Propósito |
| Nombre | String | Privada | Es el nombre del ejercicio. Funciona como identificador. |
| Tipo de ejercicio | String | Privada | Es el tipo de ejercicio al que este ejercicio pertenece. El tipo de ejercicio puede ser Aerobico, Muscular, de Flexibilidad y de Equilibrio. |
| Nivel de cansancio | String | Privada | Es el nivel de cansancio que hacer este ejercicio genera. Puede ser Ligero, Intermedio o Pesado. |
| Número de repeticiones | int | Privada | Es el número de veces que este ejercicio se debe realizar en un solo espacio de tiempo. |
| Calorías quemadas | double | Privada | Es la cantidad total de calorías que el ejercicio ayuda a quemar |

Clase: Rutina

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo de dato | Visibilidad | Propósito |
| Nombre | String | Privada | Es el nombre de la rutina de ejercicios. Funciona como identificador. |
| Lista de ejercicios | ArrayList<String> | Privada | Es la lista de ejercicios que la rutina incluye. Se incluye solo el nombre de los ejercicios, no tiene ningún dato específico de ellos. |

Clase: Contador de rutina

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo de dato | Visibilidad | Propósito |
| Nombre de rutina | String | Privada | Es el nombre de la rutina del cual el contador de ejercicios lleva la cuenta. |
| Cantidad de usos | Int | Privada | Es el número de veces que la rutina se ha asignado. |

Clase: Controlador

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo de dato | Visibilidad | Propósito |
| Lista de miembros | ArrayList<Miembro> | Privada | Es la lista de miembros que se encuentran registrados en el sistema. |
| Lista de entrenadores | ArrayList<Entrenador> | Privada | Es la lista de entrenadores que se encuentran registrados en el sistema. |
| Lista de ejercicios | ArrayList<Ejercicio> | Privada | Es la lista de ejercicios que se han registrado en el sistema. |
| Lista de rutinas | ArrayList<Rutinas> | Privada | Es la lista de rutinas que se encuentran registradas en el sistema. |

Clase: Vista

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo de dato | Visibilidad | Propósito |
| Controlador | Controlador | Privada | Es el controlador que el programa usa para realizar sus funciones. |
| Teclado | Scanner | Privada | Es el método por el cual el programa recibe texto del usuario. |

### 1.4 Métodos de cada clase (10 pts)

Clase: Miembro

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método | Parámetros (tipo de dato) | Resultado (tipo de dato) | Visibilidad | Propósito |
| Asignarse a entrenador | Nombre de entrenador: String | boolean | publica | Este método asigna al miembro de gimnasio a un entrenador. En caso ya tenga un entrenador, no se asigna el nuevo y se regresa False. De lo contrario, se asigna el nuevo entrenador y se regresa True. |
| Desasignarse de entrenador | Ninguno | boolean | Publica | Este método remueve el entrenador que el miembro de gimnasio tenía asignado. En caso de que no tuviese a ninguno asignado, regresa False. De lo contrario, remueve al entrenador y regresa True. |

Clase: Entrenador

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método | Parámetros (tipo de dato) | Resultado (tipo de dato) | Visibilidad | Propósito |
| Añadir alumno | Nombre de alumno: String | void | publica | Este método asigna a un alumno al entrenador. Se añade el nombre del alumno a la lista de alumnos a los que el entrenador enseña actualmente. |
| Quitar alumno | Nombre de alumno: String | boolean | Publica | Este método remueve a uno de los alumnos que un entrenador tiene. Se busca el nombre del alumno en la lista de alumnos que el entrenador tiene asignados, y se quita. Si no se encuentra al alumno en la lista, se regresa False. De lo contrario, se regresa True. |
| Anadir Rutina | Nombre de rutina: String | void | Publica | Este método actualiza las rutinas que un entrenador usa. Para ello busca si el entrenador ya tiene un Contador de ejercicios con el nombre de la rutina. En caso de tenerlo, le agrega 1 a su cuenta. En caso de no tenerlo, crea un nuevo Contador de ejercicios y coloca su cuenta en 1. |
| Quitar Rutina | Nombre de rutina: String | boolean | Publica | Este método remueve una de las rutinas que el entrenador tiene actualmente. Para ello busca en los Contadores de rutina que el entrenador tiene, si alguno coincide con el nombre de rutina ingresado. Si no se encontró la rutina, se regresa false. De lo contrario se quita 1 de su cuenta, y en caso de que la nueva cuenta sea 0, se quita el Contador de rutina. Al final, se regresa true. |

Clase: Ejercicio

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método | Parámetros (tipo de dato) | Resultado (tipo de dato) | Visibilidad | Propósito |
| Ninguno |  |  |  |  |

Clase: Contador de Rutinas

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método | Parámetros (tipo de dato) | Resultado (tipo de dato) | Visibilidad | Propósito |
| Ninguno |  |  |  |  |

Clase: Rutina

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método | Parámetros (tipo de dato) | Resultado (tipo de dato) | Visibilidad | Propósito |
| Añadir ejercicio | Nombre de ejercicio: String | void | Publica | Este método añade un ejercicio a la lista de ejercicios que la rutina usa. Puede ingresarse el mismo nombre varias veces, lo que indica que este ejercicio se realiza múltiples veces en la rutina. |
| Quitar ejercicio | Nombre de ejercicio: String | boolean | Publica | Este método remueve uno de los ejercicios que la lista de ejercicios de la rutina tiene. Para ello recorre la lista de ejercicios, y remueve el primer nombre del ejercicio que coincida con el ingresado. Al final, regresa True. Si se recorrió toda la lista y no se encontró el ejercicio, se regresa False. |
| Consultar ejercicios | Ninguno | String | Publica | Este método devuelve un mensaje con los ejercicios que están asignados en una rutina. Para ello, recorre la lista de ejercicios que la rutina tiene, y añade el nombre de cada ejercicio a una lista nueva. Si la lista ya contiene uno de los ejercicios, el nuevo no se añade. Al final, se convierte la lista a un String, y se devuelve dicho String. |

Clase: Controlador

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método | Parámetros (tipo de dato) | Resultado (tipo de dato) | Visibilidad | Propósito |
| Crear miembro | Nombre: String  Edad: int  Peso actual: double  Peso ideal: double  Tipo de membresía: String | void | publica | Este método crea un nuevo objeto de Miembro usando los parámetros ingresados, y lo añade a la lista de miembros del sistema |
| Crear entrenador | Nombre: String  Edad: int  Cantidad máxima de alumnos: int | void | Publica | Este método crea un nuevo objeto de Entrenador usando los parámetros ingresados, y lo añade a la lista de entrenadores del sistema. |
| Crear ejercicio | Nombre: String  Tipo de ejercicio: String  Nivel de cansancio: String  Número de repeticiones: int  Calorías quemadas: double | Void | Publica | Este método crea un nuevo objeto de Ejercicio, y lo añade a la lista de ejercicios con los que cuenta el sistema. |
| Crear rutina | Nombre de rutina: String  Lista de ejercicios: String… | void | Publica | Crea un nuevo objeto de Rutina con los parámetros dados, y lo añade a la lista de rutinas con las que cuenta el sistema. |
| Mostrar lista | Nombre de lista: String | String | Publica | Esta función recibe el nombre de una lista del sistema, y con dicha lista regresa un String con el nombre de todos los elementos que esta lista contenga. Las listas que pueden consultarse son: miembros, entrenadores, rutinas. |
| Mostrar estadísticas del sistema | ninguno | String | Publica | Calcula algunas estadísticas relacionadas con los elementos del sistema, y regresa un mensaje con los resultados. Estas estadísticas son: el entrenador con más alumnos, el entrenador con menos alumnos, que rutina es la más usada, que rutina es la menos usada y el porcentaje de entrenadores que se encuentran sobrecargados. |
| Anadir ejercicio a rutina | Nombre de ejercicio: String  Nombre de rutina: String | void | Publica | Busca entre la lista de ejercicios que el sistema tiene guardados, y al encontrar aquel que coincida con el nombre ingresado se mantiene “seleccionado”. Después de ello, se busca en la lista de rutinas del sistema, y si el nombre de una coincide con el ingresado se le asigna el ejercicio “seleccionado”. |
| Eliminar ejercicio de rutina | Nombre de ejercicio: String  Nombre de rutina: String | boolean | Publica | Busca entre la lista de rutinas que el sistema tiene guardados, y se busca una que coincida con el nombre ingresado. Si una de ellas coincide, se busca el nombre de ejercicio ingresado en la lista de ejercicios de la rutina. Si se encuentra, se elimina el objeto de Ejercicio correspondiente y se regresa True. Si no se encuentra, se regresa False. |
| Asignar miembro a entrenador | Nombre de miembro: String  Nombre de entrenador: String  Nombre de rutina: String | void | Publica | Busca entre la lista de entrenadores que el sistema tiene. Si alguno de ellos coincide con el nombre ingresado, se le asigna el nombre del miembro al entrenador y se le asigna el nombre del entrenador al usuario del sistema correspondiente. Además de esto, se le asigna al miembro la rutina que usará para entrenar, y se llama la función de AnadirRutina del entrenador. |
| Desasignar miembro de entrenador | Nombre de miembro: String  Nombre de entrenador: String | boolean | Publica | Busca en la lista de entrenadores del sistema, si alguno de ellos coincide con el nombre de entrenador ingresado. En caso de serlo, se busca en la lista de alumnos del entrenador si alguno coincide con el nombre de miembro ingresado. En caso de coincidir, se elimina el miembro de la lista de alumnos del entrenador, se baja 1 a la cuenta de la rutina que el miembro usaba, y se elimina el entrenador que el miembro tenía asignado. Al final se regresa True. En caso de no encontrarse el miembro en el entrenador, se regresa False. |
| Validar nombre existe | Nombre a validar: String  Lista en donde buscar: String | boolean | Publica | Se busca en una lista si el nobre ingresado existe o no. Las listas en donde se puede buscar si existe o no un elemento son la de Miembros, Entrenadores, Ejercicios y Rutinas. |

Clase: Vista

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método | Parámetros (tipo de dato) | Resultado (tipo de dato) | Visibilidad | Propósito |
| Main | Ninguno | void | Publica | Es el método principal del programa. En el se recorre el flujo del programa, y al terminarse se termina el programa. |
| Crear elemento | Tipo de elemento: String | void | Publica | Se encarga de recibir y validar los datos necesarios para crear un elemento de cierto tipo en el sistema. Para ello, revisa que el nombre del elemento no exista en la lista correspondiente en el sistema, y que los datos a usar sean correctos. Todos estos datos se pasan al controlador, en donde se crea el objeto necesario y se asigna a su lista correspondiente. |
| Asignar miembro a entrenador | Ninguno | void | Publica | Pide los datos necesarios para asignar a un miembro del sistema a un entrenador especifico. Para ello le pide al usuario un nombre de miembro, un nombre de entrenador y la rutina que usará. Estos datos se mandan al controlador, en donde se realizan las asignaciones correspondientes. |
| Desasignar miembro de entrenador | Ninguno | void | Publica | Pide los datos necesarios para desasignar a un miembro de un entrenador. Para ello, le pide al usuario un nombre del miembro y del entrenador. Estos datos se mandan al controlador, y se hace la desasignación correspondiente. |
| Añadir ejercicio a rutina | Ninguno | Void | Publica | Pide al usuario que ingrese el nombre de una de las rutinas que se encuentran en el sistema. Seguido de ello, le pide que ingrese el nombre de uno de los ejercicios que quisiera añadir a la rutina. Se envían estos datos al controlador, y se hace la operación correspondiente. |
| Quitar ejercicio de rutina | Ninguno | void | publica | Pide al usuario que ingrese el nombre de una de las rutinas que se encuentran en el sistema. Seguido de ello, le pide que ingrese el nombre de uno de los ejercicios que quisiera quitar de la rutina. Se envían estos datos al controlador, y se hace la operación correspondiente. |
| Consultar elemento | Ninguno | void | Publica | Se le pide un tipo de dato al usuario, y muestra todos los elementos de ese tipo que se encuentren en el sistema. Los tipos de dato a ingresar son Miembros, Entrenadores, Ejercicios y Rutinas. |
| Calcular estadisticas | Ninguno | void | Publica | Se le muestra al usuario ciertas estadísticas con las que cuenta el programa. Estas estadísticas son el entrenador y la rutina más y menos populares, y un porcentaje de cuantos entrenadores se encuentran sobrecargados. |

### 2. Diseño: Diagrama de Clases (30 puntos)

* Asegúrate de mostrar atributos y métodos con visibilidad (+, -).
* Indica relaciones entre clases (asociación, agregación, etc.).
* Incluye el driver program (Main).

<https://lucid.app/lucidchart/31f1b705-ca60-482a-885f-4a982aa4cd1b/edit?viewport_loc=-11%2C-10%2C1705%2C917%2C0_0&invitationId=inv_1b0570c9-1424-4cae-9758-29c6f65d6cac>

### 3. Programa (40 puntos)

En cada archivo `.java`, asegurarse de incluir:

* Las clases necesarias.
* Uso adecuado de objetos.

Menú que debe implementar el driver program:

* 1. Nuevo comprador
  2. Nueva solicitud de boletos
  3. …
  4. …
  5. Salir

GitHub: colocar aquí la URL:

https://github.com/\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Checklist antes de entregar

Está claro el análisis?

El diagrama tiene los elementos UML correctamente?

Subiste tu código a GitHub con todo lo necesario?