Instalaciones Deportivas

Programación II
Curso 2016/17

Convocatoria Enero

Revisión 1

Índice

Índice 3
1 Especificación del proyecto 4
1.1 Objetivo del sistema 4
1.2 Descripción del problema 4
2 Análisis y Diseño del proyecto 5
2.1 Modelos UML 5
2.2 Software UML 5
2.3 Entrega y evaluación del diseño 5
3 Implementación del proyecto 6
3.1 Almacenamiento permanente 6
3.2 Funcionamiento del programa 6
3.2.1 Insertar persona 7
3.2.2 Asignar monitor 8
3.2.3 Alta usuario 8
3.2.4 Asignar Grupo 9
3.2.5 Gestionar pago de actividades 9
3.2.6 Obtener calendario 10
3.2.7 Ordenar monitores por carga horaria de actividades 11
3.2.8 Ordenar actividades por número de usuarios dados de alta 11
3.3 Formato de ficheros 12
3.3.1 Fichero “personas.txt” 12
3.3.2 Fichero “actividades.txt” 13
3.3.3 Fichero “ejecucion.txt” 15
3.3.4 Fichero “avisos.txt” 15
3.4 Comentarios a la implementación 16
3.5 Otras restricciones de funcionamiento. 17
4 Entrega y evaluación del proyecto. 17

1 Especificación del proyecto

1.1 Objetivo del sistema

El objetivo de este proyecto será el desarrollo de una aplicación que almacene y gestione las personas que hacen uso de unas instalaciones deportivas y las actividades que ahí se desarrollan. En particular, la aplicación deberá realizar las operaciones indicadas sobre personal (monitores) y los usuarios en lo que se refiere a la asignación de actividades y grupos.

1.2 Descripción del problema

Se diseñará un programa que gestione las personas que forman parte de la vida de unas instalaciones deportivas, restringiendo estas a personal y usuarios. Todas ellas compartirán una serie de rasgos comunes y, al mismo tiempo, para su caracterización serán necesarios atributos y propiedades particulares. De esta manera, todas las personas que forman parte del sistema deberán tener un nombre y apellidos, fecha de nacimiento, y un identificador único en el sistema. Además de la información anterior, (1) para un usuario es relevante la fecha de alta y el saldo disponible en su cuenta del sistema; y (2) para un monitor la actividad en la que es especialista.

Como es natural, los monitores son aquellas personas que dirigen y supervisan actividades en las instalaciones deportivas (a las que asisten los usuarios). Por otro lado, la oferta deportiva incluye un conjunto de actividades, caracterizadas por su nombre, sus siglas así como el conjunto de actividades previas que se requiere haber realizado anteriormente. Dependiendo del número de usuarios en cada actividad, la misma se reparte en uno o más grupos de usuarios. Cada actividad se desarrolla en unas instalaciones determinadas: piscina, pabellón, etc.

El proceso básico en el las instalaciones deportivas es el desarrollo de actividades. A este respecto, cada monitor tiene una dedicación semanal en horas, que determina el máximo número de horas semanales que puede dedicar a ejercer su profesión.

Respecto a los usuarios, estos se dan de alta en distintas actividades y luego seleccionan (o se les asigna) uno de los grupos de dicha actividad. A este proceso de asignación se aplican dos restricciones. Primero, un usuario sólo puede matricularse en una actividad si ha superado todas las actividades previas requeridas. Segundo, no se permite a un usuario darse de alta en una actividad y seleccionar un grupo que se solapa en el horario con cualquier otro grupo en el que esté inscrito.

Con este escenario, se pretende desarrollar una aplicación que proporcione la funcionalidad necesaria para gestionar y consultar las listas de personas y actividades según lo indicado en este documento. El usuario de esta aplicación desempeñará el rol de gestor de las instalaciones y tendrá acceso a consultar/añadir/modificar todos aquellos datos que considere oportuno según los comandos previstos en este documento. El número de personas y actividades

en el sistema no tiene límite a priori. Tanto las personas como las actividades tendrán identificadores únicos.

2 Análisis y Diseño del proyecto

2.1 Modelos UML

La primera parte del proyecto consistirá en el diseño de un modelo de clases adecuado para el proyecto.

Diagrama de clases. A partir del análisis de la descripción en este documento, el alumno debe elaborar un diagrama de clases (de análisis) que identifique las principales clases y relaciones en el dominio del problema. Este diagrama será refinado para la obtención de un diagrama detallado de clases (de diseño) que:

- Incluya los atributos, multiplicidades, navegaciones que sea posible extraer de la descripción de la funcionalidad.
- Identifique la semántica de las relaciones: agregación, composición, nombres de rol, etc.
- Utilice la herencia (relación de generalización) y las interfaces (relación de realización) si es necesario.
- Detalle los métodos principales que permitan aportar la funcionalidad descrita en los modelos de casos de uso. (NOTA: deberán incorporarse todos los métodos necesarios para cubrir la funcionalidad, no sólo los *getters* y *setters*).

En resumen, se elaborarán dos diagramas de clases:

- Uno de alto nivel, sin métodos por clase.
- Otro detallado o de bajo nivel, que contenga los métodos incorporados en cada clase/interface.

2.2 Software UML

El alumno podrá utilizar cualquier herramienta (*plugin* o *standalone*) conforme con UML para la realización del diseño. Se recomienda, no obstante, usar la herramienta *Visual Paradigm for UML*, que puede descargarse de la página web *http://www.visual-paradigm.com/download/*.

2.3 Entrega y evaluación del diseño

El diseño del proyecto se realizará en grupos de dos alumnos, siendo los grupos los mismos grupos constituidos al comienzo del curso en las sesiones B.

La entrega, se realizará a través de la plataforma FaiTIC, utilizando un único fichero PDF que documente el diseño realizado. Se entregará un único fichero por grupo. El documento de diseño debe contener los diagramas de clases de alto y bajo nivel.

En el proceso de revisión y valoración por parte del profesor del diseño UML de cada grupo, el profesor **podrá** solicitar la defensa del mismo por parte de los

miembros del grupo, que se realizará en sesiones de grupo B y será de asistencia obligatoria para ambos miembros del grupo. La puntuación máxima del diseño del proyecto será de 1 punto.

3 Implementación del proyecto

3.1 Almacenamiento permanente

El conjunto de personas con el que trabaja el programa se almacenará en un fichero, que se denominará "personas.txt" y que tendrá el formato indicado en el apartado 3.3 de este documento. Asimismo, la información de las actividades se almacenará en un fichero denominado "actividades.txt" cuyo formato también se define en ese mismo apartado.

3.2 Funcionamiento del programa

La gestión de información se realizará de manera no interactiva, es decir, el gestor ejecutará el programa y este deberá realizar todas las operaciones precisas para completar las órdenes indicadas en un fichero de invocaciones, y sin volver a requerir ningún dato, finalizar la ejecución de un modo completo y ordenado. Para ello se definen los archivos de datos según un formato específico que será obligatorio respetar en todos los detalles explicitados y un archivo de invocaciones que también deberá respetarse para indicar al sistema las operaciones que de él se van a ejecutar esperan.

Cuando se invoque el programa, consultará un fichero denominado "ejecucion.txt" que contendrá las operaciones que deberá llevar a cabo y los parámetros necesarios para completar dichas operaciones. Este archivo podrá contener una o más operaciones que deberán realizarse en el mismo orden en el que se presentan en este fichero de entrada.

Cada línea de este archivo (que no sea un comentario) se corresponderá con una invocación independiente. De esta manera, aún cuando se produzca una situación que genere un aviso del sistema y se aborte la ejecución del comando en cuestión, será necesario continuar con la ejecución del siguiente comando si lo hubiera en este archivo. La ejecución de cada comando usará los datos que resultasen de la invocación anterior, fuera ésta con el resultado que fuera.

En caso de que se produzca algún error en la invocación o ejecución de lo solicitado, el programa deberá añadir al fichero "avisos.txt" una línea con el texto exacto indicado en esta memoria. Este archivo, en caso de no existir, deberá ser creado por el propio programa. En caso de producirse algún otro error, que si bien sea factible, no esté mencionado en esta memoria, deberá obviarse. **En caso de que sí se den varios de los avisos explicitados, todos deben ser recogidos y reflejarse en el mismo orden en el que aparecen en esta memoria.** En caso de que se produzca alguno de estos avisos, la ejecución del programa deberá mantener los datos preexistentes en el sistema.

Cada invocación de una operación a realizar ocupará exactamente una línea de dicho fichero. Su formato será específico para cada operación y se detalla en las secciones siguientes. En caso de que el comando introducido no exista se deberá generar, en el archivo de avisos, el aviso "Comando incorrecto: <comando>" donde <comando> es el texto concreto del comando en el archivo de ejecución. **Sin embargo, no se considerará error el hecho de que entre los argumentos de un comando pueda haber más de un espacio en blanco. Asimismo, el nombre de cada comando podrá ir indistintamente en minúsculas o mayúsculas.**

3.2.1 Insertar persona

Este comando deberá introducir un nuevo usuario o monitor en el sistema. El formato para la línea correspondiente al fichero de ejecuciones será el siguiente:

```
InsertaPersona perfil nombre apellidos fecha1 [fecha2]
[saldo] [horasAsignables]
```

En esta **sintaxis**, indicar que:

- *perfil*, se corresponde con la descripción del tipo de personal que se considera en este sistema. Podrá tomar uno de los siguientes valores: monitor o usuario.
- *nombre*, se corresponde con el nombre. Irá delimitado por comillas dobles (") ya que puede contener espacios.
- *apellidos*, se corresponde con los apellidos. Irá delimitado por comillas dobles (") ya que puede contener espacios.
- *fecha1*, se corresponde con la fecha de nacimiento y seguirá el formato dd/mm/aaaa.
- *fecha2* (para el caso de usuarios), se corresponde con la fecha de ingreso en las instalaciones y seguirá el formato dd/mm/aaaa.
- *saldo*(para el caso de usuario), cantidad disponible en la cuenta del usuario. Será un número decimal (sólo dos decimales) que indique una cantidad en euros.
- *horasAsignables* (para el caso de monitores), indicará cuantas horas semanales se le pueden asignar, como máximo, para la realización de sus tareas. Será un número entero.
-

El identificador asociado a la nueva persona será el primer entero positivo libre (excluyendo el 0) en el momento de realizar la inserción.

Los **avisos** que puede generar este comando serán:

- *"Fecha incorrecta*", si alguna de las fechas *fecha1* o *fecha2* son incorrectas en si mismas, o no están comprendidas entre 01/01/1950 y 01/01/2020.
- (para usuarios) "*Fecha de ingreso incorrecta*", si el tiempo entre la fecha de nacimiento y la de ingreso es menor de 5 años o mayor de 80.
- (para usuarios) "*Saldo incorrecto"*, en caso de que el saldo introducido sea negativo.
- (para monitores) "*Numero de horas incorrecto*", en caso de que las horas asignadas sean menores que 0 o mayores que el máximo (ver punto 3.3.1).

3.2.2 Asignar monitor

Este comando deberá asignar un monitor a un grupo de una actividad. El formato para la línea correspondiente al fichero de ejecuciones será el siguiente:

```
AsignarMonitorGrupo persona actividad grupo
```

En esta **sintaxis**:
- *persona*, se corresponde con el identificador del monitor que se considera.
- *actividad*, se corresponde con el identificador de la actividad.
- *grupo*, se corresponde con el identificador del grupo que finalmente se le asigne.

Los **avisos** que puede generar este comando serán:

- *"Monitor inexistente"*, en caso de que no exista ningún monitor con el identificador indicado.
- "*Actividad inexistente*", en caso de que no exista ninguna actividad con el identificador indicado.
- *"Grupo inexistente"*, en caso de que sí exista la actividad indicada pero no exista el grupo solicitado.
- *"Grupo ya asignado"*, en caso de que se intente reasignar un grupo ya asignado.
- *"Horas asignables superior al máximo"*, en caso de que el valor de las horas asignables sea mayor del máximo del monitor.
- *"Se genera solape"*, en caso de que el nuevo grupo genere algún tipo de solape temporal con las asignaciones previas a ese monitor.

3.2.3 Alta usuario

Este comando deberá enrolar a un usuario en una actividad. El formato para la línea correspondiente al fichero de ejecuciones será el siguiente:

```
Alta usuario actividad
```

En esta **sintaxis**:
- *usuario*, se corresponde con el identificador del usuario a dar de alta.
- *actividad*, se corresponde con el identificador de la actividad en consideración.

Los **avisos** que puede generar este comando serán:

- "*Usuario inexistente*", en caso de que no exista ningún usuario con el identificador indicado.
- "*Actividad inexistente*", en caso de que no exista ninguna actividad con el identificador indicado.
- "*Ya es usuario de la actividad indicada*", en caso de que se intente asignar a una actividad de la que ya forma parte.
- "*No cumple requisitos*", en caso de que el usuario sí exista pero no tenga las actividades previas realizadas.

3.2.4 Asignar Grupo

Este comando deberá asignar un grupo de una actividad a un usuario que previamente ha sido asignado a ella. El formato para la línea correspondiente al fichero de ejecuciones será el siguiente:

```
AsignaGrupo usuario actividad grupo
```

En esta **sintaxis**:

- *usuario*, se corresponde con el identificador del usuario a asignar.
- *actividad*, se corresponde con el identificador de la actividad en consideración.
- *instalación,* indica donde se va a desarrollar el grupo (valores "Piscina1", "Piscina2", "Pabellon1", "Pabellon2").
- *grupo*, se corresponde con el identificador del grupo que se le asigne.

Los **avisos** que puede generar este comando serán:

- "*Usuario inexistente*", en caso de que no exista ningún usuario con el identificador indicado.
- "*Actividad inexistente*", en caso de que no exista ninguna actividad con el identificador indicado.
- "*Usuario no dado de alta en la actividad*" en caso de que no se haya dado de alta previamente en la actividad.
- "*Grupo inexistente*", en caso de que sí exista la actividad indicada pero no exista el grupo solicitado.
- *"Se genera solape"*, en caso de que el nuevo grupo genere algún tipo de solape temporal con las asignaciones previas de la misma u otras actividades.

3.2.5 Gestionar pago de actividades

Este comando deberá permitir saldar el pago de las actividades y modificar de modo consecuente toda la información en el sistema. El formato para la línea correspondiente al fichero de ejecuciones será el siguiente:

```
Cobrar usuario ficheroActiv
```

En esta **sintaxis**:

- *usuario*, se corresponde con el identificador de la usuario en consideración
- *ficheroActiv*, es el nombre del fichero que contendrá las actividades a cobrar, con un único dato por cada línea que será el id de la actividad a cobrar.

El programa deberá procesar el fichero, y para cada línea deberá:

1.- Comprobar si el usuario está dado de alta en la actividad indicada en el *ficheroActiv* (aunque no tenga un grupo asignado).

2.- En caso positivo descontar el valor del saldo del usuario en función del coste de la actividad.

3.- En caso que el usuario no tenga saldo suficiente para acometer el pago de todas sus actividades se realizará el pago de todas las actividades posibles manteniendo el saldo positivo o cero y se generará un error por cada una de las que no se pueda pagar.

El fichero ficheroActiv quedará inalterado, en cualquier caso.

Los **avisos** que puede generar este comando serán:

- "*Usuario inexistente*", en caso de que no exista ningún usuario con el identificador indicado.
- "*Fichero de actividades inexistente*", en caso de que "*ficheroActiv*" no exista.

Para las líneas de actividad del fichero, se podrán generar los errores siguientes:

- "*Actividad no existente*" en caso de que no exista la actividad.
- "*Saldo insuficiente para pagar la actividad: <Id Actividad>*" , en caso de que no haya saldo suficiente para pagar esta actividad.

Cada uno de estos posibles errores irá precedido del texto "Error en línea <nº línea>:" que informe de la línea del fichero en la que se produce el error. La primera línea del fichero corresponde a la línea "1".

Se procesarán todas las líneas del fichero *ficheroActiv*. Es decir, si alguna de ellas tuviese algún error se generaría el aviso correspondiente y se pasaría a procesar la siguiente línea.

3.2.6 Obtener calendario

Este comando deberá generar el horario de actividades de un usuario. El formato para la línea correspondiente al fichero de ejecuciones será el siguiente:

```
ObtenerCalendario usuario salida
```

En esta **sintaxis**:

- *usuario*, se corresponde con el identificador del usuario en consideración.
- *salida*, se corresponde con el nombre del fichero en el que deberá quedar almacenada la salida del programa. Este fichero deberá contener una única línea por cada grupo, con la sintaxis siguiente:

  L/M/X/J/V/S/D; hora; instalación; id grupo; nombre de la actividad; nombre y apellidos del instructor que imparte el grupo.

  Separándose cada campo por punto y coma (;), y deberá ordenarse por orden cronológico creciente (se puede asumir que no coinciden).

La primera línea del fichero será para un encabezado de los campos conteniendo el texto: “dia; hora; instalación; id grupo; actividad; instructor “.

Los **avisos** que puede generar este comando serán:

- “*Usuario inexistente*”, en caso de que no exista ningún usuario con el identificador indicado. En este caso no debe generarse el fichero de salida.

- “*Usuario sin asignaciones”*, en caso de que sí exista el usuario, pero este no tenga ningún grupo activo. En este caso tampoco debe generarse el fichero de salida.

3.2.7 Ordenar monitores por carga horaria de actividades

Este comando deberá generar un listado de monitores ordenados por carga horaria de actividades asignada, es decir, aquella que resulta de computar los grupos en los que esté asociado. El formato para la línea correspondiente al fichero de ejecuciones será el siguiente:

```
OrdenarMonitoresPorCarga salida
```

En esta **sintaxis**:

- *salida*, se corresponde con el nombre del fichero en el que deberá quedar almacenada la salida del programa. Este fichero deberá seguir la sintaxis del fichero “personas.txt” y estará ordenado por orden creciente de carga horaria. En caso de empate se recurrirá al identificador del monitor (yendo primero el de menor identificador).

El **aviso** que puede generar este comando es:

- “*No hay monitores*”, en caso de que no haya ningún monitor titular, En este caso no se debe generar el fichero de salida.

3.2.8 Ordenar actividades por número de usuarios dados de alta

Este comando deberá generar un listado de las actividades ordenadas por el número de usuarios que estén en cada una. El formato para la línea correspondiente al fichero de ejecuciones será el siguiente:

```
OrdenarActividades salida
```

En esta **sintaxis**:

- *salida*, se corresponde con el nombre del fichero en el que deberá quedar almacenada la salida del programa. Este fichero deberá seguir la sintaxis del fichero “actividades.txt” y estará ordenado por orden creciente de número de usuarios. En caso de empate se recurrirá al identificador de la actividad(yendo

primero el de menor identificador). Si una actividad no tiene ningún usuario , no deberá aparecer en el listado.

El **aviso** que puede generar este comando es:

- "*No hay ningún usuario*", en caso de que no haya ningún usuario en ninguna actividad, En este caso no se debe generar el fichero de salida.

3.3 Formato de ficheros

El programa trabaja con los ficheros siguientes para el almacenamiento de datos:

1. Fichero "personas.txt", que almacena de forma permanente la información sobre todas las personas gestionadas en este proyecto.

2. Fichero "actividades.txt", que almacena de forma permanente la información sobre las actividades consideradas.

3. Fichero "ejecución.txt", que almacena los comandos a ejecutar.

4. Fichero "avisos.txt", que almacena el resultado de la ejecución de cada comando.

El fichero de avisos puede no existir antes de la invocación del comando, pero los otros tres mencionados sí existirán siempre al invocar al programa. El formato que obligatoriamente deben seguir estos ficheros se detalla en los siguientes apartados.

3.3.1 Fichero "personas.txt"

Este fichero contendrá los datos de las personas incluidas en el sistema. Estas personas están caracterizadas por un bloque de datos comunes y otro bloque dependiente de su perfil (usuario, monitor).

Cada uno de los datos que identifican cada persona se guardará en una línea distinta y siempre en el mismo orden. Solo existirán aquellas líneas que sean pertinentes según el tipo de persona del que se trate. Cada entrada en este fichero se separará de la siguiente mediante una línea en el fichero que contendrá solamente el carácter "*". Los datos que identificarán cada persona serán los siguientes (y se almacenarán en el fichero por este orden):

Campo	Valores
Identificador único: ID_Persona	Entero
Tipo	{usuario, monitor}
Nombre	String de, como máximo, 50 caracteres

Apellidos	String de, como máximo, 50 caracteres
Fecha de nacimiento	Con formato dd/mm/aaaa
Fecha de ingreso (usuario)	Con formato dd/mm/aaaa
Actividades Previas (usuario)	ID_Actividad, ID_Actividad, ID_Actividad, … *(tantas como actividades previas).*
Saldo (usuario)	Real positivo (con dos decimales)
Horas asignables semanales (monitor)	Entero
Actividades Actuales (usuario) *(un par por grupo asignado, habrá tantos pares como grupos a los que asista usuario)*	ID_Acividad [ID_Grupo]; ID_Actividad [ID_Grupo]; ID_Actividad [ID_Grupo]; …
Grupo impartido (monitor) *(habrá tantas triplas como grupos imparta el monitor)*	ID_Actividad ID_Grupo; ID_Actividad ID_Grupo; ID_Actividad ID_Grupo; …

donde:

- El *ID_Persona* será un identificador único generado y gestionado internamente por el sistema.
- *Tipo*, será una cadena de caracteres de entre el conjunto *{usuario, monitor}.*
- *Nombre,* el nombre (simple o compuesto) de la persona.
- *Apellidos,* los apellidos uno o varios de la persona.
- *Fecha de Nacimiento,* en el formato dd/mm/aaaa.
- *Fecha de ingreso (Para usuario),* fecha de ingreso en el formato dd/mm/aaaa.
- *Actividades Realizadas (Para usuario),* es una lista de los identificadores de actividades (*ID_Usuario*) previamente. *ID_Actividad* se define en 3.3.2.
- *Saldo (Para usuario),* el saldo en euros disponible del usuario.
- *Horas Asignables Semanales (Para monitor)*, es el número de horas de dedicación semanales de un monitor (en la impartición de grupos de actividades). Será de 20 como máximo.
- *Actividades actuales (Para usuario)*, son los grupos de las actividades en las que está dado de alta un usuario. Para cada uno, *ID_Actividad* es el identificador de la actividad(definido en 3.3.2) e *ID_Grupo* el identificador del grupo (definido en 3.3.2). En el caso de que el usuario esté dado de alta en una actividad, pero no tenga grupo asignado, los campos Instalación e ID_grupo no existirán.
- *Actividades impartidas (Para Monitor)*, son los grupos en los que participa un monitor. Para cada uno, *ID_Actividad* es el identificador de la actividad(definido en 3.3.2) e *ID_Grupo* el identificador del grupo (definido en 3.3.2).

3.3.2 Fichero “actividades.txt”

Este fichero guardará información sobre la oferta actual, esto es, la identificación de las actividades así como los grupos impartidos de cada actividad. Las actividades tienen un identificador único (asignado de forma interna por el sistema).

Campo	Valores

Identificador de actividad (*ID_Actividad*)	Entero
Nombre	String de, como máximo, 75 caracteres
Siglas	String de, como máximo, 10 caracteres
Coordinador	ID del monitor coordinador
Pre-requisitos	ID_Actividad, ID_Actividad, ID_Actividad
Duración	Entero, con valores 1 o 2.
Coste	Valor decimal indicando el coste en euros de esta actividad
Grupos	ID_1 Día_1 Hora_1 Instalación_1; ID_2 Día_2 Hora_2 Instalación_2; ID_3 Día_3 Hora_3 Instalación_3; … *(Tantas cuaternas separadas por ; como grupos se impartan de una actividad)*

En los campos anteriores:

- *Id Actividad* es el identificador único de cada actividad, un entero positivo.
- *Nombre*: es un nombre simple o compuesto para la actividad, con un máximo de 75 caracteres.
- *Siglas*: una abreviatura (no compuesta) para la actividad, de, como máximo, 10 caracteres.
- *Pre-requisitos.* Son los identificadores de las actividades que es obligatorio haber realizado para poder apuntarse a esta. Puede ser una lista variable de 0 a un máximo de 10.
- *Coste:* Coste de la actividad en euros con dos decimales.
- *Duración:* Duración de los grupos de esa actividad: 1 o 2 horas.
- *Grupos*, indica el número de horas de los grupos de cada actividad en particular y la instalación en la que se desarrollan (“Piscina1”, “Piscina2”, “Pabellon1”, “Pabellon2”). Todos los grupos de una actividad tienen la misma duración. Los valores permitidos serán de 1 o 2 horas.

Donde, para la especificación de los grupos impartidos:

- ID, será un identificador numérico único para cada grupo en una actividad.
- *Día,* toma sus valores en {L, M, X, J, V, S, D} para la identificación del día de la semana: Lunes, Martes, Miércoles, Jueves, Viernes, Sábado y Domingo, respectivamente.
- *Hora,* es la hora de comienzo de impartición de dicho grupo. Toma sus valores en el rango [9-23]. Las actividades se organizan en sesiones de 1 o 2 horas semanales. No es posible planificar una sesión antes de las 9:00 ni después de las 23:00 debido a los horarios de apertura de las instalaciones.
- *Instalacion:,* lugar donde se realiza el grupo, string de máximo 20 caracteres.

El número de líneas por actividad será siempre 8. Cada bloque de líneas relacionadas con una actividad estará separada del siguiente mediante un línea con un asterisco ("*").

Se facilitan archivos de ejemplo en FaiTIC que se recomienda consultar.

3.3.3 Fichero "ejecucion.txt"

Este archivo, estará formado por una o más líneas de texto, con el siguiente formato, por cada linea:

- Numero de línea
- Comando. Sera alguno de los comandos a implementar descritos anteriormente.

Numero de línea y comando deberán estar separados por uno o mas espacios en blanco.

Aspectos a tener en cuenta sobre este archivo:

- Si el archivo no existe al comenzar la ejecución del programa se generará el error "Fichero de ejecución no existente" y se saldrá del programa.
- El fichero de ejecución deberá permitir líneas de comentario, que serán todas aquellas que comiencen con el caracter '*'.
- Este archivo no se modificará durante la ejecución del programa.

3.3.4 Fichero "avisos.txt"

Este archivo no tiene necesariamente que existir al invocar al sistema. En él se irán almacenando de modo consecutivo las líneas correspondientes a los avisos generados en cada comando. El contenido de cada línea constara de tres partes y en ese orden:

1. El número de línea correspondiente del fichero de ejecución.
2. Una abreviatura del comando que lo generó, de acuerdo con la siguiente tabla:

Comando	Abreviatura
Inserta Persona	IP
Asignar monitor	AMON
Alta usuario	AUSER
Asignar grupo	AGRUPO
Gestionar Cobro	GCOBRO
Obtener Calendario	OCALEN

Ordenar monitores	OMON
Ordenar Actividades	OACT
Cuando no se identifique con ningún comando (Ej: Comando Incorrecto)	En blanco

3. El texto del aviso indicado por cada comando.

separándose el campo 1 del 2 y el 2 del 3 por " -- " (espacio-guion-guion-espacio).

Si el fichero ya existe al realizar la invocación, los nuevos avisos se añadirán al final del mismo (sin eliminar el contenido ya existente).

3.4 Comentarios a la implementación

- Se puede suponer que el contenido de los ficheros de actividades y personas con los que trabaja el programa siempre es correcto. El fichero de ejecución podrá, sin embargo, contener errores (es decir, faltarle algún argumento a algún comando, contener un comando inexistente, etc.).

- A la finalización de la ejecución de los comandos del archivo ejecución (contenga éste varios o un único comando), los ficheros de actividades, personas y avisos deberán quedar adecuadamente modificados, reflejando las modificaciones derivadas de cada uno de los comandos del archivo.

- El código desarrollado deberá incluir **obligatoriamente** la información necesaria para generar la documentación de dicho código en formato Javadoc.

- Cualquier aclaración que el profesorado de la asignatura considere necesario realizar sobre este enunciado se comunicará a través de la plataforma FaiTIC y será de obligado cumplimiento.

3.5 Otras restricciones de funcionamiento.

Cara a la implementación se deberán tener en cuenta las siguientes restricciones:

1. Ids de grupo por grupo y actividad. Los ids de los grupos serán por actividad.
2. Un grupo solo podrá ser impartido por un único monitor.

3. Una actividad deberá tener al menos un grupo. Un usuario solo podrá tener asignado un máximo de un grupo de cada actividad.

4. Un monitor puede no tener actividades asignadas.

5. Un usuario solo podrá tener asignado un *máximo* de un grupo de cada actividad.

6. El máximo de actividades que podrá coordinador un monitor será de 2.

7. El número máximo de usuarios por grupo es de 20.

4 Entrega y evaluación del proyecto.

En evaluación continua, y para los grupos de 2 personas, la evaluación del proyecto será por parejas, debiendo presentarse ambos miembros de cada pareja. En el caso de que uno de los miembros no se presente la puntuación del no presentado será de 0 puntos.

Tras la evaluación del diseño UML, cada grupo continuará conjuntamente con la implementación del proyecto. **Antes del 5 de enero a las 23:00**, cada grupo subirá a la plataforma FaiTIC un único fichero en formato ZIP que incluya la carpeta Eclipse del proyecto. El nombre del fichero deber ser **entrega#.zip** (donde # indica el número del grupo de Pii); siendo obligatorio usar archivos con el formato zip. El fichero compilado deberá encontrarse también en el primer nivel de directorios (directamente accesible al descomprimir el fichero) y deberá llamarse obligatoriamente `Proyecto#.class` (donde # indica el número del grupo de Pii) .

El resultado de la evaluación del proyecto tendrá en cuenta:

- Correcto funcionamiento (CF), (evaluado de 0 a 2.8 puntos) determinado por una serie de pruebas que se le pasarán al proyecto. Estas pruebas se publicarán en FaiTIC, en algún momento con posterioridad a la fecha de entrega del proyecto.

- Calidad del código (evaluado de 0.5 a 1 puntos) evaluado de acuerdo a los patrones de diseño orientado a objetos (DOO), contemplándose aspectos como:

    - Correcto modelado de la información.
    - Correcta selección y ubicación de métodos en clases.
    - Uso de los principios básicos del diseño en Java (herencia, polimorfismo, sobrecarga, interfaces, sobre-escritura, etc.)
    - Legibilidad del código.
    - Estructuración y modularidad del código.

- Mejoras o características opcionales (evaluado hasta un máximo de 0,4 puntos) incorporados por el grupo (OPC). Se considerarán mejoras las relacionadas en el punto 4.1. Serán también susceptibles otras mejoras propuestas por el grupo, pero éste debe **validarlas**, *antes* de implementarlas, con el profesor, quien le asignará la puntuación adecuada.

La implementación será calificada con la nota obtenida según el cálculo: (CF+OPC)*DOO, (hasta un máximo de 3 puntos).

NOTA IMPORTANTE.
La evaluación del proyecto se realizará en los equipos de los laboratorios docentes.

4.1 Mejoras evaluables.

- *Ordenar usuarios por saldo disponible*. (0.2 puntos). Producirá una salida a fichero de todos los alumnos ordenados por el saldo disponible, de mayor a menor. Por cada línea se imprimirán tres datos: Nombre de usuario(apellidos, nombre), dni y saldo. En caso de que haya dos usuarios con el mismo saldo los ordenara por Apellido y Nombre. El formato del comando será:

    OrdenaUsuariosXSaldo salida

- *Crear actividad*. (0.3 puntos). Añadir un comando nuevo para crear una actividad con al menos un grupo y añadirla al fichero de actividades, manteniendo el formato del archivo.

- *Utilización del paquete swing*. (0.4 puntos). Dos opciones:

    - Presentación en forma tabular del contenido del fichero de personas (diferenciando usuarios y monitores) por dni, del archivo de actividades por siglas y del archivo de avisos por fecha.

    - Presentación en forma jerárquica y tabular del saldo de los usuarios.