



CARÁTULA

Tema	Grupo	Ayudante	Correcciones			
			Fecha	Hora Inicio	Hora Fin	Resultado
			Entrega			
			Revisión			

INTEGRANTES

	Padrón	Apellido y Nombre	Asistencia a Entrega	Asistencia a Revisión	Evaluación Individual Final
1					
2					
3					
4					
5					
6					

PLANILLA DE EVALUACIÓN – camino feliz

ITEM	OK	NoK	Inc	Observaciones
Entrega				
Trae carpeta (hojas sujetas)?				
Trae readme impreso?				
Presentación				
Caratulas completas (dos)				
Índice, pie, nro. de hoja				
Documenta hipótesis y problemas relevantes?				
Instalación				
Trae dispositivo para arranque desde puerto USB?				
Trae paquete completo?				
en README: da indicaciones para Booteo, logueo?				
Instruye sobre cómo descargar/desempaquetar?				
Da instrucciones sobre cómo instalar el tp?				
Crea ok el archivo de configuración?				
Ejecución CIPAK – Camino Feliz				
Hay hoja de ruta del camino feliz documentado?				
Identifica los archivos usados en el camino feliz?				
Logra completar el camino feliz?				



PrepararAmbiente				
Setea variables?				
Setea permisos?				
Dispara RecibirOfertas?				
Graba log?				
RecibirOfertas				
Lee los archivos de ARRIDIR?				
Contabiliza los ciclos?				
Valida el nombre del archivo?				
Mueve los aceptados en OKDIR?				
Mueve los rechazados en NOKDIR?				
Dispara ProcesarOfertas?				
Graba Log?				
GenerarSorteo				
Graba el archivo de sorteos?				
Graba Log?				
ProcesarOfertas				
Valida el contrato, el Grupo?				
Valida el importe, mínimo, máximo?				
Determina bien si participa o no?				
Graba ofertas validas?				
Mueve archivos a PROCDIR?				
Graba Log?				
Cierran ok los totales?				
DeterminarGanadores				
Muestra opción de ayuda				
Calcula OK Ganadores por Sorteo?				
Calcula OK Ganadores por Licitación?				
Graba ok?				
FUNCIONES				
LanzarProceso				
DetenerProceso				
MoverArchivos				
GrabarBitacora				
MostrarBitacora				



Índice

Enunciado – Tema L	4
Introducción Narrativa	4
Evaluación	5
Documentación Solicitada	6
Contenido de la Carpeta	6
Especificación de comandos y Funciones	8
Recomendaciones para el equipo de desarrollo	8
Indicaciones para el equipo de instalación	8
Indicaciones para el equipo de integración y testing	10
PrepararAmbiente	11
RecibirOfertas	13
ProcesarOfertas	15
GenerarSorteo	18
DeterminarGanadores	19
A. Resultado general del sorteo	20
B. Ganadores por sorteo	20
C. Ganadores por licitación	20
D. Resultados por grupo	21
Función LanzarProceso	22
Función DetenerProceso	22
Función MoverArchivos	23
Función GrabarBitacora	24
Función MostrarBitacora	25
Estructuras y Archivos	26
Maestro de Concesionarios: MAEDIR /concesionarios.csv	26
Tabla de Fechas de Adjudicación: MAEDIR /FechasAdj.csv	26
Archivos de ofertas: ARRIDIR / <cod_concesionario> _<animesdia>.csv	26
Padrón de suscriptores: MAEDIR /temaL_padron.csv	26
Tabla de Grupos: MAEDIR /Grupos.csv	27
Archivo de ofertas válidas: PROCDIR /validas/<fecha_de_adjudicacion>.txt	27
Archivo de ofertas rechazadas: PROCDIR /rechazadas/<cod_concesionario>.rech	28
Archivos de sorteos PROCDIR /sorteos/<Id_Sorteo>_<fecha_adj>.srt	28
Archivos de Log: LOGDIR / <nombre del comando> .log	28



Enunciado – Tema K

Introducción Narrativa

Una empresa que vende unidades automotoras a través del sistema de plan de ahorro previo para fines determinados desea crear un simulador de adjudicaciones por sorteo y licitación para evaluar el reemplazo del sistema actual

Todos los meses, se realiza un acto de adjudicación según las dos modalidades previstas: Sorteo o Licitación.

Licitación consiste en ofertar una suma de dinero a criterio de cada suscriptor dentro de un monto mínimo y máximo establecido por la Sociedad Administradora. La unidad por licitación será adjudicada al contrato con mayor monto ofertado en cada uno de los grupos.

Para licitar se debe presentar una oferta en sobre cerrado en las oficinas de cualquier concesionario oficial

El concesionario crea un archivo de ofertas y las remite con la frecuencia que crea conveniente para que este simulador las procese.

El sistema valida las ofertas, determina los participantes y la posición obtenida.

En caso de igualdad de monto de dos o más ofertas de licitación, se determinará el ganador según el orden establecido en el Sorteo electrónico que se realiza en el acto de adjudicación.

El Sorteo consiste en la generación de números aleatorios sin repetición del 1 al 168.

En cada acto de adjudicación se realiza un sorteo.

Ese sorteo se aplica a todos los grupos administrados.

Al sistema lo llamaremos **CIPAK** y el mismo estará compuesto por:

1. La documentación del sistema **CIPAK**

Es parte de la resolución del TP la entrega de una carpeta con la Documentación Solicitada.

2. Un comando Shell **PrepararAmbiente** para la configuración del entorno de ejecución

El Proceso se inicia con el aseguramiento de la disponibilidad de la información para llevar adelante el proceso total.

Continúa con la asignación de valor a un conjunto de variables de ambiente.

Finalmente ofrece arrancar automáticamente el comando **RecibirOfertas**.

3. Un comando shell **RecibirOfertas** para la recepción de los archivos de ofertas

En cada archivo viene información de varias ofertas, realizadas en varios días

Si el nombre del archivo (filename) cumple con el formato de nombre esperado y el archivo es texto, el archivo se acepta, de lo contrario se lo rechaza.

Cuando verifica que hay archivos aceptados, arranca, si corresponde, **ProcesarOfertas**.

4. Un comando Shell **ProcesarOfertas** para determinación de participantes de la licitación

Los registros de ofertas validas se graban en una nueva estructura de archivo empleada luego para determinar ganadores

5. Un comando Shell **GenerarSorteo** para determinación del sorteo

6. Un comando PERL **DeterminarGanadores** para la generación de listados de ganadores por sorteo y licitación

Se requiere que estos comandos trabajen en forma integrada, no deben ser comandos independientes ya que la naturaleza del TP es que desarrollen UN SISTEMA.

7. Funciones Complementarias



-
- Una Función (en Shell o en Perl) denominada MoverArchivos que se emplea para mover archivos
 - Una Función (en Shell o en Perl) denominada GrabarBitacora que se emplea para grabar los archivos de log
 - Una Función (en Shell o en Perl) denominada MostrarBitacora que se emplea para buscar en los archivos de log
 - Función en Shell script denominada DetenerProceso que se emplea para detener procesos y otra complementaria LanzarProceso que permite disparar proceso.

Evaluación

El día de vencimiento del TP, cada ayudante convocará a los integrantes de un grupo, solicitará la carpeta y el paquete de instalación e iniciará la corrección mediante una entrevista grupal.

Es imprescindible la presencia de todos los integrantes del grupo el día de la corrección

Se evaluará el trabajo grupal y a cada integrante en forma individual. El objetivo de esto es comprender la dinámica de trabajo del equipo y los roles que ha desempeñado cada integrante del grupo.

Para que el alumno apruebe el trabajo práctico debe estar aprobado en los dos aspectos: grupal e individual.

Dentro de los ítems a chequear el ayudante evaluará aspectos formales (como ser la forma de presentación de la carpeta), aspectos funcionales: que se resuelva el problema planteado y aspectos operativos: que el TP funcione integrado.

El paquete de instalación se deberá remitir vía correo electrónico a so7508@gmail.com. En el asunto del correo indicar Nro. de Grupo y Ayudante asignado. En el cuerpo del correo se debe indicar la versión de Sistema Operativo usada para la ejecución.



Documentación Solicitada

Contenido de la Carpeta

La carpeta para entregar el día de la corrección al ayudante designado debe contener los siguientes elementos:

1. Carátula

La entregada en este mismo documento con los datos completos en 2 COPIAS una para el grupo y otra para el docente.

2. Planillas de Evaluación

Las entregadas en este mismo documento

3. Índice del Contenido de la Carpeta.

Con número de página en cada ítem, el mismo puede ser incorporado manualmente

4. Hipótesis y Aclaraciones Globales

Documente las hipótesis que ha considerado para la resolución del TP. Documente cualquier otra aclaración que se considere necesaria. Todas las hipótesis deben presentarse en este punto, puede agruparlas por comando.

5. Problemas relevantes

Describa los problemas relevantes que se hayan presentado durante el desarrollo, la integración y/o la prueba del sistema. Explique cómo fueron solucionados.

6. Archivo README

Incluya la impresión del README en la carpeta. Ver detalles de su contenido en "Contenido del README"

7. Listado de Nuevas Funciones y/o Comandos Auxiliares

Brinde un listado de las nuevas funciones y/o comandos auxiliares creados por Ustedes, es decir, que no figuran en el enunciado original del TP.

Indique: Nombre de la función, quienes la usan, para que la usan.

Si no crea ninguna, indique: NINGUNA

8. Listado de DATOS

Imprima un listado con todos los nombres de los archivos de datos entregados por la cátedra y su cantidad de registros

9. Listado de Nuevos Archivos

Brinde un listado de los nuevos archivos creados por Ustedes, es decir, que no figuran en el enunciado original del TP.

Indique: Nombre del archivo, si es temporal o permanente, donde lo almacenan, quienes lo usan, para que lo usan. Si no crea ninguno, indique: NINGUNO

10. Hoja de ruta de prueba "camino feliz"

- Instale el tp e Imprima el contenido del archivo de configuración
- Imprima las primeras 10 líneas de contenido de los archivos maestros
- Ejecute [PrepararAmbiente](#) y permita que el demonio arranque. Imprima el log de [PrepararAmbiente](#)
- Tome dos archivos de ofertas que tengan nombres aceptables y deposítelos en ARRIDIR.
- Espere



- Imprima el log de [RecibirOfertas](#) y los primeros 10 registros de cada uno de los archivos aceptados.
- Imprima el log de [ProcesarOfertas](#) y los primeros 10 registros de cada uno de los archivos de resultado.
- Ejecute [GenerarSorteo](#) e imprima el archivo de salida
- Ejecute [DeterminarGanadores](#) con la opción de ayuda
- Ejecute [DeterminarGanadores](#) con la opción provisoria
- Ejecute [DeterminarGanadores](#) con la opción definitiva y de grabación
- Imprima las invocaciones y los distintos resultados obtenidos.

11. Apéndice, este mismo documento

Incluya como apéndice este documento sin la carátula ni las planillas de evaluación dado que ya fueron incluidas al principio de la carpeta

La documentación debe entregarse en una carpeta con TODAS las hojas numeradas y sujetas. Las hojas sueltas no se considerarán como parte de la misma.

El pie de página de cada hoja debe tener: Número de Grupo y Tema (en el margen izquierdo) y Número de Hoja (en el margen derecho). La numeración puede ser manual.



Especificación de comandos y Funciones

Recomendaciones para el equipo de desarrollo

1. Se deberá tener en cuenta para la resolución TODAS las condiciones que se enuncian.

Se deben respetar los formatos de archivos especificados y la estructura de directorios planteada

Los pasos de ejecución sugeridos son solo a los efectos de ordenar la explicación, por lo cual deben considerarse meramente indicativos.

Si el equipo de desarrollo lo considera pertinente, puede modificarlos tanto sea en el orden de ejecución como en la forma de resolverlo, siempre y cuando esto no afecte el resultado final esperado. También pueden:

- Crear nuevos scripts
- Aumentar la funcionalidad de los comandos solicitados

Estos cambios deben estar documentados en la carpeta que entrega el día de vencimiento del tp.

2. Archivos Auxiliares

Se debe evitar el uso de archivos auxiliares permanentes, los archivos auxiliares temporales, se deben eliminar ANTES de finalizar la ejecución del comando.

3. Movimiento de Archivos

En líneas generales no se borra ningún archivo de datos, se los mueve de un lugar a otro para asegurar la integridad de la información original. Se solicita una función de librería [MoverArchivos](#) para el movimiento de archivos de datos la cual debe ser empleada por todos los comandos que la requieran

4. Manejo de errores, logueo

Toda invocación desde un comando a otro debe devolver un código de retorno cero (0) si fue exitoso o distinto de cero si tuvo errores.

Todo evento que genera algún tipo de error debe ser grabado en el log y mostrado por pantalla

EVITE retrasar el proceso de evaluación/corrección del TP debido a la falta de rastreo de eventos. Es por ello que se recomienda dejar en los scripts las pistas de rastreo que crea convenientes condicionadas a un flag que se enciende si es necesario. Esto evitara que en la ejecución estándar se inunde de mensajes sin interés para el usuario.

La escritura en el TODOS los archivos de log debe ser homogénea, es por ello que se centraliza en la función [GrabarBitacora](#)

5. Archivo de Configuración

La instalación deja un archivo de configuración con varios registros con el contenido de variables usadas en el sistema. Los desarrolladores pueden agregar más registros si lo consideran necesario.

En cada registro de este archivo se define una variable del sistema.

Indicaciones para el equipo de instalación

1. El paquete de instalación deberá estar contenido en un único archivo instalable en formato ".tgz" con todos los archivos y directorios empaquetados en un archivo "tar" y luego comprimido con "gzip". El instalable deberá contener:

- La documentación, los scripts desarrollados y los archivos con datos (maestros y novedades) entregados por la cátedra
- Los casos de prueba creados por el grupo bien identificados. Pueden incluir en la documentación información del caso: nombre del archivo y que lo caracteriza (que es lo que se prueba con ese archivo)

2. Directorio de Trabajo

- Para realizar la instalación el directorio de trabajo debe ser Grupoxx, donde xx es su número de grupo



- Todo el camino (path) que va desde la raíz hasta Grupox lo denominaremos genéricamente en esta explicación \$GRUPO

3. Estructura de \$GRUPO

Como resultado de la instalación la estructura de \$GRUPO debe ser la siguiente:

- \$GRUPO/binarios en donde se depositarán los scripts ejecutables, al cual genéricamente denominaremos: BINDIR
- \$GRUPO/maestros en donde depositarán los archivos maestros, al cual genéricamente denominaremos MAEDIR
- \$GRUPO/arribados para simular la recepción de archivos de novedades, al cual genéricamente denominaremos: ARRIDIR
- \$GRUPO/datos con al menos los archivos de prueba entregados por la cátedra
- \$GRUPO/aceptados para depositar los archivos aceptados, al cual genéricamente denominaremos: **OKDIR**
- \$GRUPO/procesados para depositar los Archivos de ofertas Procesadas, al cual genéricamente denominaremos **PROCDIR**
- \$GRUPO/informes para depositar los Reportes, al cual genéricamente denominaremos **INFODIR**
- \$GRUPO/bitacoras para depositar los Archivos de Log, al cual genéricamente denominaremos LOGDIR
- \$GRUPO/rechazados para depositar los Archivos Rechazados, al cual genéricamente denominaremos **NOKDIR**
- \$GRUPO/config para depositar el archivo de configuración, al cual genéricamente denominaremos CONFDIR

4. Archivo de Configuración

La instalación debe dejar un archivo donde se registra la estructura de \$GRUPO y otras variables del sistema

El nombre de este archivo es CIPAK.cnf, su separador de campos es el signo igual “=”

Cada registro de este archivo tiene los siguientes campos:

Campo	Descripción/Fuente/Valor	
Variable	Caracteres	Nombre de la variable
Valor	Caracteres	Valor asignado a la variable
Usuario	Caracteres	Es el login del usuario que graba el registro
Fecha	Fecha y hora	Formato a Elección. Es la fecha y hora en el momento de grabación del registro.

Las variables a grabar son:

- GRUPO
- BINDIR
- MAEDIR
- ARRIDIR
- **OKDIR**
- **PROCDIR**
- **INFODIR**
- LOGDIR



- **NOKDIR**
- LOGSIZE
- SLEEPTIME

Ejemplo: GRUPO=/usr/grupo23=Sandra=09/04/2016 10:03 p.m

5. Contenido del README

Como parte de la documentación del sistema se debe proveer la impresión del README indicando booteo, logueo, descarga del paquete, instalación del paquete, arranque del sistema, detención/arranque de procesos, etc. Por ejemplo, se debe dar:

- Una explicación de cómo arrancar el SO desde puerto USB
- Una explicación de cómo loguearse
- Una explicación de cómo descargar el paquete
- Una explicación de cómo copiar, descomprimir, crear directorio del grupo, etc
- Una explicación sobre que se requiere para poder instalar y/o ejecutar el sistema
- Instrucciones de instalación del sistema [CIPAK](#)
- Que nos deja la instalación y dónde
- Cuáles son los primeros pasos para poder ejecutar el sistema
- Como arrancar o detener comandos
- Cualquier otra indicación que considere necesaria

Indicaciones para el equipo de integración y testing

1. La carpeta de entrega del TP y el paquete de instalación INCLUYEN la demostración de que llegaron a ejecutar el camino feliz.
2. La ejecución no debe ser de un comando AISLADO del siguiente, sino integrado, partiendo de un conjunto de novedades HASTA lograr la emisión de reportes con esas novedades procesadas

Debido a esto deben realizar la integración antes de la entrega del TP para poder subsanar los errores de comunicación que surjan entre los comandos encadenados.

3. La cátedra provee los archivos maestros necesarios para la ejecución del sistema

Si los archivos de prueba remitidos por la cátedra contienen caracteres incompatibles con su configuración (por ejemplo el carácter de fin de registro o fin de archivo), puede realizar la conversión que necesite siempre que sea homogénea (igual para todos los archivos) y sin modificar los datos

Si lo hace, indíquelo en el punto hipótesis y aclaraciones globales de la carpeta

4. También se proveen archivos de prueba (novedades) con un alto porcentaje de información libre de error, es responsabilidad del equipo de testing generar otros archivos de prueba con casos lo suficientemente heterogéneos como para contemplar todas las variantes de ejecución, en particular las de rechazo o error



PrepararAmbiente

Input

- Archivo de Configuración **CONFDIR/CIPAK.cnf**
- Ejecutables **BINDIR/***
- Maestros **MAEDIR/***
- Directorio de Resguardo a definir por el instalador

Output

- Log del Comando **LOGDIR/PrepararAmbiente.log**

Opciones y Parámetros

- A especificar por el desarrollador

Descripción

El propósito de este comando es preparar el ambiente del sistema y dejarlo listo para su ejecución.

- Es el primero en orden de ejecución
- Se dispara manualmente
- Graba en el archivo de Log a través del [GrabarBitacora](#)
- **Repara, si corresponde, la instalación**
- Invoca, si corresponde, el siguiente proceso: [RecibirOfertas](#)

El Proceso se inicia con el aseguramiento de la disponibilidad de la información para llevar adelante el proceso total: Es indispensable contar con el archivo de configuración, los comandos y los archivos maestros todos ellos con los permisos adecuados.

Continúa con la asignación de valor a un conjunto de variables de ambiente que van a ser usadas por el resto del sistema y luego ofrece arrancar automáticamente el comando [RecibirOfertas](#)

El resto de los comandos JAMAS deben acceder al archivo de configuración para conocer directorios o el contenido de las variables del sistema, deben usar las variables de ambiente que define este proceso.

No se puede ejecutar ningún comando si la inicialización de ambiente no fue realizada

Indicar en las Hipótesis Globales donde se realiza este control: en el programa llamador o en el llamado.

Pasos Sugeridos

1. Verificar si el ambiente ya ha sido inicializado.

[PrepararAmbiente](#) debe setear las variables de ambiente una sola vez por cada sesión de usuario.

Si se intenta ejecutar mas de una vez en la misma sesión de usuario, no permitirlo y explicar la situación: Por ejemplo indicar

```
Ambiente ya inicializado, para reiniciar termine la sesión e ingrese nuevamente
```

Grabar en el log y terminar la ejecución.

2. Verificar que la instalación está completa

Este control debe contemplar que existan en el directorio de ejecutables todos los scripts y en el directorio de maestros todos los archivos maestros.



Si se detecta algún faltante explicar la situación con un mensaje que indique los componentes faltantes

2.1. Si es posible reparar la instalación, explicar con un mensaje que se procederá a hacerlo

Realizar la reparación de la instalación copiando donde corresponda los archivos faltantes desde el repositorio de resguardo

2.2. Si no es posible reparar la instalación dar indicaciones para que un administrador lo haga y Terminar la ejecución.

3. Verificar los permisos

Si se detecta que algún archivo no tiene los permisos adecuados explicar la situación con un mensaje

Configurar los permisos de los archivos correctamente

Si no se puede efectuar la corrección, mostrar mensaje explicativo y Terminar la ejecución

4. Inicializar el ambiente

Setear la variable PATH y cualquier otra variable de ambiente que considere necesarias, como ser:

- GRUPO
- BINDIR
- MAEDIR
- ARRDIR
- **OKDIR**
- **PROCDIR**
- **INFODIR**
- LOGDIR
- **NOKDIR**
- LOGSIZE
- **SLEEPTIME**, etc.

5. Mostrar y Grabar en el log el siguiente mensaje

Estado del Sistema: **INICIALIZADO**

6. Mostrar y Grabar en el log todas las variables con su contenido.

7. Preguntar si se desea arrancar RecibirOfertas

PrepararAmbiente debe ofrecer la posibilidad de arrancar el demonio **RecibirOfertas**

Para ello se debe preguntar al operador por ejemplo mostrando el siguiente mensaje:

¿Desea efectuar la activación de **RecibirOfertas**? Si - No

7.1. Si el usuario no desea arrancar el demonio **RecibirOfertas**, entonces explicar cómo hacerlo con el comando **LanzarProceso**

7.2. Si el usuario desea arrancar el demonio **RecibirOfertas**, activarlo (SOLO SI NO EXISTE OTRO **RecibirOfertas** CORRIENDO) y explicar cómo detenerlo usando el comando **DetenerProceso**.

7.3. Mostrar mensaje y grabar en el log

RecibirOfertas corriendo bajo el no.: <Process Id de **RecibirOfertas**>

8. FINAL:

Cerrar el archivo de log- Terminar el proceso



RecibirOfertas

Input

- Concesionarios **MAEDIR**/concesionarios.csv
- Fechas de Adjudicación **MAEDIR**/FechasAdj.csv
- Archivos de Input **ARRIDIR**/`<cod_concesionario>_<aniomesdia>`.csv

Output

- Archivos Aceptados **OKDIR**/`<nombre del archivo>`
- Archivos Rechazados **NOKDIR**/`<nombre del archivo>`
- Log del Comando **LOGDIR**/RecibirOfertas.log

Descripción

El propósito de este comando es detectar la llegada de archivos al directorio **ARRIDIR** y aceptar o rechazar estos archivos según corresponda

- Es el segundo en orden de ejecución
- Es un proceso del tipo "Demonio" :
- Se dispara con [PrepararAmbiente](#) o a través del [LanzarProceso](#)
- Se detiene a través del [DetenerProceso](#)
- Mueve los archivos a través del [MoverArchivos](#)
- Graba en el archivo de Log a través del [GrabarBitacora](#)
- Invoca, si corresponde, el siguiente proceso: [ProcesarOfertas](#)

El Proceso se inicia con la detección de la presencia de archivos en el directorio **ARRIDIR**

Si el nombre del archivo (filename) cumple con el formato de nombre esperado y el archivo es de texto, el archivo se acepta, de lo contrario se lo rechaza.

También verifica si hay archivos ya aceptados para arrancar automáticamente el proceso [ProcesarOfertas](#)

Luego duerme un tiempo [SLEEPTIME](#) y vuelve a a empezar, es decir, que a menos que se detenga con [DetenerProceso](#), este proceso no tiene condición de fin.

A este tipo de programas se los denomina demonio, daemon o dæmon (de sus siglas en inglés Disk And Execution Monitor).

Otra característica de los procesos del tipo demonio, es que se ejecutan en segundo plano en vez de ser controlado directamente por el usuario (es un proceso no interactivo).

SLEEPTIME es una variable de ambiente seteada por el proceso [PrepararAmbiente](#). Su valor lo toma del archivo de configuración, el cual se genera con la instalación

Se debe mantener un contador de ciclos de ejecución del [RecibirOfertas](#).

Recuerde que no se puede ejecutar ningún comando si la inicialización de ambiente no fue realizada

Indicar en las Hipótesis Globales donde se realiza este control: en el programa llamador o en el llamado.

Pasos Sugeridos:

INICIO

1. Grabar en el Log el nro de ciclo:

```
RecibirOfertas ciclo nro. 1
```



2. Chequear si hay archivos en el directorio **ARRIDIR**

Si existen archivos, continuar en el siguiente paso. Si no existen archivos ir al paso NOVEDADES PENDIENTES?

UN ARCHIVO

3. Verificar que el archivo sea un archivo común, de texto.

Los archivos de cualquier otro tipo, se rechazan.

4. Verificar que el formato del nombre del archivo sea correcto

FORMATO CORRECTO: <cod_concesionario>_<aniomesdia>.csv

Los archivos con nombres que no se correspondan con el formato esperado, se rechazan.

5. Validar el nombre del los archivos:

- Cod_concesionario debe existir en el maestro de concesionarios
- ANIOMESDIA debe ser una fecha válida
- ANIOMESDIA debe ser menor o igual a la fecha del día y mayor a la fecha del último acto de adjudicación

6. Aceptar los archivos con nombre válido.

Si el nombre del archivo es válido mover el archivo aceptado a **OKDIR**/**<nombre del archivo>** empleando la función **MoverArchivos**

Grabar en el log el mensaje de archivo aceptado con el nombre y el path correspondiente

7. Rechazar los archivos inválidos

- Si el nombre del archivo NO es válido, rechazarlo
- Si el archivo viene vacío, rechazarlo
- Si el archivo no es un archivo común, de texto (si es una imagen, un comprimido, etc), rechazarlo

Para rechazar un archivo moverlo a **NOKDIR** empleando la función **MoverArchivos**

Grabar en el log: nombre del archivo y cuál ha sido el motivo del rechazo, por ejemplo

- Tipo de archivo invalido
- fecha invalida
- fecha fuera de rango
- concesionario inexistente
- cualquier otro error que considere pertinente indicar.

NOVEDADES PENDIENTES?

8. Ver si existen archivos en **OKDIR**

Si existen (de este ciclo o de un ciclo anterior), invocar al Comando **ProcesarOfertas** siempre que éste no esté corriendo

- Si se pudo Invocar, grabar en el log :

ProcesarOfertas corriendo bajo el no.: <Process Id de **ProcesarOfertas**>

- Si correspondía invocar pero se debe posponer, grabar en el log:

Invocación de **ProcesarOfertas** pospuesta para el siguiente ciclo

- Si da algún tipo de error grabar el mensaje explicativo

9. Dormir un tiempo **SLEEPTIME** y Volver al INICIO



ProcesarOfertas

Input

- Archivos de Ofertas **OKDIR**/`<cod_concesionario>_<aniomesdia>.csv`
- Padrón de Suscriptores **MAEDIR**/temaK_padron.csv
- Tabla de Fechas de adjudicación **MAEDIR**/fechas_adj.csv
- Tabla de Grupos **MAEDIR**/grupos.csv

Output

- Archivo de ofertas validas **PROCDIR/validas**/`<fecha_de_adjudicación >.txt`
- Archivos procesados **PROCDIR/procesadas**/`<nombre del archivo>`
- Archivos de ofertas rechazadas **PROCDIR/rechazadas**/`<cod_concesionario>.rech`
- Archivos rechazados (archivo completo) **NOKDIR**/`<nombre del archivo>`
- Log del Comando **LOGDIR**/ProcesarOfertas.log

Opciones y Parámetros

- A especificar por el desarrollador

Descripción

El propósito de este comando es reunir en un solo archivo todas las ofertas validas que participaran del acto de adjudicación

- Es el tercero en orden de ejecución
- Es invocado por [RecibirOfertas](#)
- Graba las ofertas validas en un archivo de salida
- Graba las ofertas rechazadas indicando el motivo de rechazo
- Graba en el archivo de Log a través del [GrabarBitacora](#)
- Mueve los archivos a través del [MoverArchivos](#)
- No debe procesar dos veces un mismo archivo

Licitación consiste en ofertar una suma de dinero a criterio de cada suscriptor dentro de un monto mínimo y máximo establecido por la Sociedad Administradora.

Los montos mínimos y máximos dependen del grupo, del valor de la cuota pura y de la cantidad de cuotas que restan hasta terminar el plan de ahorro.

De acuerdo al grupo, el monto mínimo es igual a valor de cuota pura * cantidad de cuotas para licitación y el monto máximo es igual a valor de cuota puta * cantidad de cuotas pendientes

Pasos sugeridos

1. Procesar todos los archivos

El orden de procesamiento de los archivos debe hacerse cronológico desde el antiguo al más reciente según sea la fecha que figura en el nombre del archivo

Inicializar el log grabando:

```
Inicio de ProcesarOfertas  
Cantidad de archivos a procesar:<cantidad>
```

Los archivos de input se encuentran en **OKDIR**

2. Procesar Un Archivo

Procesar un archivo es procesar todos los registros que contiene ese archivo, excepto cuando el archivo ya fue procesado o bien no posee la estructura interna adecuada. En estos dos casos corresponde el rechazo del archivo completo.



2.1 Verificar que no sea un archivo duplicado

Cada vez que se procesa un archivo, se lo mueve tal cual fue recibido y con el mismo nombre a **PROCDIR/procesadas**

Es por ello que es posible detectar antes de intentar procesar un archivo si ya fue procesado solo inspeccionando el contenido de ese directorio. Si ya fue procesado, rechazar el archivo completo y grabar en el log un mensaje aclaratorio, como ser:

```
"Se rechaza el archivo por estar DUPLICADO".
```

El archivo duplicado se lo mueve a **NOKDIR** empleando la función **MoverArchivos**.

2.2 Verificar la cantidad de campos del primer registro

Si la cantidad de campos del primer registro no se corresponde con el formato establecido, asumir que el archivo está dañado, rechazar el archivo completo y grabar en el log un mensaje aclaratorio, como ser:

```
"Se rechaza el archivo porque su estructura no se corresponde con el formato esperado".
```

El archivo rechazado se lo mueve a **NOKDIR** empleando la función **MoverArchivos**

3. Si se puede procesar el archivo

Grabar en el log

```
Archivo a procesar: <nombre del archivo a procesar>
```

4. Validar oferta

Campo	Descripción/Fuente/Valor
Contrato Fusionado = Grupo y Numero de Orden	7 caracteres. Los primeros 4 caracteres corresponden al número de grupo, los siguientes 3 corresponden al Numero de orden del suscriptor dentro del grupo El contrato Fusionado se debe validar contra el padrón de suscriptores MAEDIR/temaK_padron.csv (existe o no existe) El Grupo se debe validar contra el archivo de Grupos: MAEDIR/Grupos.csv (Estado del grupo ABIERTO o NUEVO)
Importe de la oferta	Importe. Mayor o igual al monto mínimo (valor de cuota pura * cantidad de cuotas para licitación) y menor o igual al monto máximo (valor de cuota pura * cantidad de cuotas pendientes)
Participa?	Flag. Es un flag del Padrón de suscriptores que sirve para determinar si el suscriptor está habilitado a participar o no del acto de adjudicación. Si en el padrón de suscriptores el suscriptor tiene la marca de participación en 1 o 2 puede participar. Si esta en blanco, no puede participar.

Si el registro NO supera estas validaciones ir a RECHAZAR REGISTRO, sino continuar

Motivos de rechazo:

- Contrato no encontrado
- No alcanza el monto Mínimo
- Supera el monto máximo
- Suscriptor no puede participar
- Grupo CERRADO

5. GRABAR oferta valida

Si las validaciones fueron ok, Grabar un registro en el Archivo de ofertas validas:



Campo	Descripción /Fuente/Valor
Código de Concesionario	Código de concesionario proveniente del nombre del archivo
Fecha del archivo	Fecha del nombre del archivo, formato a elección
Contrato Fusionado	proveniente del archivo de ofertas
Grupo	Primeros 4 caracteres del Contrato
Nro de Orden	Últimos 3 caracteres del Contrato
Importe Ofertado	Importe proveniente del archivo de ofertas
Nombre del Suscriptor	Apellido y nombre del suscriptor, proveniente del padrón de suscriptores
usuario	Login del usuario que graba el registro
fecha	Fecha y hora de grabación del registro, en el formato que se desee

La fecha de adjudicación del nombre del archivo a grabar se obtiene de la Tabla de Fechas de adjudicación **MAEDIR/fechas_adj.csv** y se corresponde con la fecha del próximo acto de adjudicación

Incrementar los contadores adecuados

Continuar con el siguiente registro.

6. RECHAZAR REGISTRO

Si alguna de las validaciones da error, rechazar el registro grabándolo en Archivo de ofertas rechazadas

Campo	Descripción /Fuente/Valor
Fuente	Nombre del archivo de input
Motivo	Motivo por el cual se rechaza ESTA oferta
Registro de Oferta	Registro Original COMPLETO
usuario	Login del usuario que graba el registro
fecha	Fecha y hora de grabación del registro rechazado, en el formato que se desee

Incrementar los contadores adecuados

Continuar con el siguiente registro.

7. Fin de Archivo

Para evitar el reprocesamiento de un mismo archivo, mover el archivo procesado a:

PROCDIR/procesadas empleando la función **MoverArchivos**.

Cuando se termina de procesar un archivo se debe grabar en el Log total de:

- Registros leídos = aaa: cantidad de ofertas validas bbb cantidad de ofertas rechazadas = ccc

8. Llevar a cero todos los contadores de registros

9. Continuar con el siguiente archivo

10. Repetir hasta que se terminen todos los archivos.

11. Fin Proceso

Grabar en el log la cantidad de archivos procesados y la cantidad de archivos rechazados

Cerrar el log grabando "Fin de **ProcesarOfertas**"



GenerarSorteo

Input

- Tabla de Fechas de adj. **MAEDIR/fechas_adj.csv**

Output

- Archivos de sorteos **PROCDIR/sorteos/<sorteoid>_<fecha de adjudicación >**
- Log del Comando **LOGDIR/GenerarSorteo.log**

Opciones y Parámetros

- A especificar por el desarrollador

Descripción

El propósito de este comando es generar números aleatorios sin repetición del 1 al 168

- Es invocado manualmente a través del comando [LanzarProceso](#)
- Graba el resultado del sorteo en un archivo de salida
- Graba en el archivo de Log a través del [GrabarBitacora](#)
- Sorteoid es un número de secuencia que permite individualizar cada uno de los sorteos que se ensayan en [GenerarSorteo](#). Es útil cuando se realiza más de un sorteo para la misma fecha de adjudicación
- Inicializar el log grabando:

Inicio de Sorteo

- Obtener el resultado del sorteo y Grabar los 168 registros del archivo de sorteo con: Nro de Orden (desde el 1 al 168 de forma creciente) y Nro de sorteo (resultado aleatorio obtenido)

Ejemplo:

Numero de orden 1	le corresponde el numero de sorteo	138
Numero de orden 2	le corresponde el numero de sorteo	60
Numero de orden 3	le corresponde el numero de sorteo	29
Numero de orden 4	le corresponde el numero de sorteo	37
Numero de orden 5	le corresponde el numero de sorteo	167
Numero de orden 6	le corresponde el numero de sorteo	36
Numero de orden 7	le corresponde el numero de sorteo	92
Numero de orden 8	le corresponde el numero de sorteo	17
Numero de orden 9	le corresponde el numero de sorteo	6
Numero de orden 10	le corresponde el numero de sorteo	32

- La fecha de adjudicación del nombre del archivo a grabar se obtiene de la Tabla de Fechas de adjudicación **MAEDIR/fechas_adj.csv** y se corresponde con la fecha del próximo acto de adjudicación
- Finalizar el log grabando:

Fin de Sorteo

Debido al alto grado de libertad que se les permite en el desarrollo de este comando, documentarlo detalladamente



DeterminarGanadores

Input

- Padrón de Suscriptores **MAEDIR/temaK_padron.csv**
- Tabla de Grupos **MAEDIR/grupos.csv**
- Archivo de ofertas validas **PROCDIR/validas/<fecha_de_adjudicación>.txt**
- Archivos de sorteos **PROCDIR/sorteos/<sorteoid>_<fecha de adjudicación>**

Output

- Resultado general del sorteo **INFODIR/<sorteoid>_<fecha de adjudicación>.txt**
- Ganadores por Sorteo **INFODIR/<sorteoid>_Grdxxxx-Grhyyyy_<fecha_adj>**
- Ganadores por Licitacion **INFODIR/<sorteoid>_Grdxxxx-Grhyyyy_<fecha_adj>**
- Resultados por grupo **INFODIR/<sorteoid>_Grupoxxxx_<fecha_adj>**

Opciones

- -a (ayuda)
- -g (grabar)

Parámetros

- Sorteo Id (obligatorio)
- Grupos
 - Un grupo
 - Varios grupos
 - Un rango de grupos
 - Todos los grupos

Tipos de consultas

- A. Resultado general del sorteo
- B. Ganadores por sorteo
- C. Ganadores por licitación (muestra los ganadores por licitación), parámetro grupos
- D. Resultados por grupo (muestra los ganadores por sorteo y los ganadores por licitación), parámetro grupos

Descripción

- Se dispara manualmente
- No graba en el archivo de log
- **DeterminarGanadores** No debe ejecutar si la inicialización de ambiente no fue realizada
- **DeterminarGanadores** No debe ejecutar si ya existe otro comando **DeterminarGanadores** en ejecución
- **DeterminarGanadores** No debe ejecutar si no existe archivo de Sorteo

Requisitos:

- Debe estar programado en PERL
- Se deben emplear estructuras Hash en la resolución



- Debe presentar un menú amigable y una **opción (-a)** de ayuda del comando
- Debe permitir al usuario efectuar N consultas sin salir del comando
- Siempre que se emplea la **opción (-g)** se debe grabar el resultado en un archivo.
- El parámetro grupo debe permitir consultar por
 - Un grupo
 - Varios grupos
 - Un rango de grupos
 - Todos los grupos
- El cálculo de ganadores se realiza solo para los grupos NO CERRADOS
- El cálculo de ganadores (tanto por sorteo como por licitación) se realiza solo para los suscriptores habilitados a participar del acto de adjudicación (marca de participación en 1 o 2)
- Siempre mostrar en forma clara y amigable la información resultante por pantalla

A. Resultado general del sorteo

Muestra el contenido del archivo de sorteos especificado como parámetro de forma amigable y ordenada por número de sorteo.

Ejemplo:

Nro. de Sorteo 001, le correspondió al número de orden 033

B. Ganadores por sorteo

Es obligatorio recibir como parámetro el Sorteo_Id

El archivo de sorteo correspondiente debe existir.

Para el o los grupos pasados como parámetro mostrar y/o grabar (ordenado por grupo) el ganador del sorteo, es decir, el suscriptor participante que obtuvo el menor número de sorteo según su número de orden dentro del grupo.

Quiénes participan?: los integrantes del grupo con el flag Participa? En 1 o 2

Título: Ganadores del Sorteo <sorteo Id> de fecha <fecha_adj>

Ganador por sorteo del grupo <gggg> Nro de Orden: <ooo>, <Nombre del Suscriptor> (Nro de Sorteo sss)

“Ganador por sorteo del grupo 7890: Nro de Orden 067, MORALES,ESTEBAN (Nro.de sorteo 023)”

C. Ganadores por licitación

Es obligatorio recibir como parámetro el Sorteo_Id

El archivo de sorteo correspondiente debe existir.

Para el o los grupos pasados como parámetro mostrar y/o grabar (ordenado por grupo) el ganador de la licitación, es decir, el suscriptor participante que ofertó el importe más alto. En caso de igualdad de monto de dos o más ofertas de licitación, se determinará el ganador según el orden establecido en el Sorteo

ATENCION: Si el ganador del sorteo también resulta ganador de la licitación, el ganador de la licitación es el suscriptor que salió en segunda posición (se debe desestimar la oferta realizada por el ganador del sorteo)



Ejemplo: grupo 7890	\$ de oferta	Numero de Sorteo	orden sin considerar sorteo	orden considerando sorteo
Numero de orden 4	73001	37	2	1
Numero de orden 23	73001	117	2	2
Numero de orden 67	74001	23	1	ganador por sorteo, automaticamente se retira la oferta por licitacion
Numero de orden 77	70500	110	5	4
Numero de orden 113	73001	165	2	3
Numero de orden 128	65000	98	6	5
Numero de orden 160	60000	80	7	6
Numero de orden 167	55000	74	8	7

Título: Ganadores por Licitación <sorteo Id> de fecha <fecha_adj>

Ganador por licitación del grupo <gggg>: Numero de orden <ooo>, <Nombre del Suscriptor> con \$nnnnnn (Nro de Sorteo sss)

Ej: "ganador por licitación del grupo 7890: numero de orden 004, FRANCO,DANIEL con \$73.001 (Nro de Sorteo 037)"

D. Resultados por grupo

Es obligatorio recibir como parámetro el Sorteo_Id

El archivo de sorteo correspondiente debe existir.

Para el o los grupos pasados como parámetro mostrar y/o grabar (ordenado por grupo) primero el ganador del sorteo (marcado con una "S") y luego el ganador por licitación (marcado con una "L")

Título: Ganadores por Grupo en el acto de adjudicación de fecha <fecha_adj>, Sorteo: <Sorteo_Id>

<Nro de Grupo>-<Nro de orden> S (<Nombre del Suscriptor>)

<Nro de Grupo>-<Nro de orden> L (<Nombre del Suscriptor>)

"7890-067 S (MORALES,ESTEBAN)

7890-004 L (FRANCO,DANIEL)"



Función [LanzarProceso](#)

Opciones y Parámetros

- parámetros u opciones a especificar por el desarrollador

Descripción

Esta función tiene por objeto disparar procesos. Es complementaria a [DetenerProceso](#).

Requerimientos

- que pueda ser invocada desde la línea de comando para arrancar cualquier proceso
 - en este caso se debe mostrar el resultado del arranque por pantalla
- que pueda ser invocada dentro de un script para arrancar cualquier proceso
 - en este caso debe registrar en el log del comando el resultado del arranque
- **Explicar claramente su invocación, parámetros u opciones y uso en el README.**

Premisas:

- No se puede arrancar un proceso si la inicialización de ambiente no fue realizada.
- No se puede arrancar un proceso si éste ya se encuentra corriendo.
- **Indicar en las Hipótesis Globales donde se realizan estos controles (en la función o en el llamador)**

Función [DetenerProceso](#)

Opciones y Parámetros

- parámetros u opciones a especificar por el desarrollador

Descripción

Esta función tiene por objeto detener procesos. Es complementaria a [LanzarProceso](#)

Requerimientos

- que pueda ser invocada desde la línea de comando para detener el demonio
- que muestre el resultado de su uso en pantalla
- **Explicar claramente su invocación, parámetros u opciones y uso en el README.**



Función *MoverArchivos*

Opciones y Parámetros

- Parámetro 1 (obligatorio): origen
- Parámetro 2 (obligatorio): destino
- Parámetro 3 (opcional): comando que la invoca
- Otros parámetros u opciones a especificar por el desarrollador

Descripción

Esta función tiene por objeto centralizar el movimiento de archivos que deben realizar la mayor parte de los comandos de este sistema para evitar diferentes políticas en el tratamiento de archivos duplicados.

En líneas generales el sistema no borra ningún archivo, los mueve de un lugar a otro, en el contexto de este TP se deben CONSERVAR todos los archivos aun cuando:

- sea un archivo improcesable, roto, dañado, vacío;
- sea un archivo con un nombre incorrecto, con espacios, mal formado;
- ya haya sido procesado
- al moverlo nos encontremos que ya existe otro archivo del mismo nombre en ese lugar.

Requerimientos

- Mover el archivo solicitado al directorio indicado sin alterar su contenido
- Si en el destino ya existe otro archivo con el mismo nombre (nombre de archivo duplicado), no debe fracasar la operación, la función debe poder conservar ambos.
 - Crear un subdirectorio /dpl para depositar el archivo duplicado y moverlo allí
 - Si también en /dpl ya existe otro archivo con el mismo nombre, emplear un número secuencial (desde 1 hasta n) para modificar el nombre del archivo y ponerlo como complemento final de la extensión: Ejemplo: <nombre del archivo original>.nnn donde nnn es el número de secuencia
- Si esta función es invocada por un comando que graba en un archivo de log, registrar el resultado de su uso en el log del comando

Premisas:

- No se puede definir un archivo auxiliar solo para registrar el número de secuencia, usar el archivo de configuración del sistema para estos propósitos
- Si el origen y el destino son iguales, no mover y registrar en el log el error
- Si el origen no existe, no mover y registrar en el log el error
- Si el destino no existe, no mover y registrar en el log el error
- Indicar en las Hipótesis Globales cuantos números de secuencia emplea: uno por directorio o uno para toda la instalación



Función GrabarBitacora

Opciones y Parámetros

- Parámetro 1 (obligatorio): comando
- Parámetro 2 (obligatorio): mensaje
- Parámetro 3 (opcional): tipo de mensaje
- Otros parámetros u opciones a especificar por el desarrollador

Descripción

¿Qué es un Log?

Un log es un registro oficial de eventos durante un periodo de tiempo en particular.

Es usado para registrar información sobre cuándo, quién, dónde, qué y por qué un evento ocurre para una aplicación, proceso o dispositivo. Es empleado por los profesionales de IT, auditoría y seguridad informática.

A estos 5 valores se los llama estándar W5, por su origen en ingles: when, who, where, what and why

- WHO: ¿Quién?
 - Usuario, es el login del usuario
- WHEN: ¿Cuándo?
 - Fecha y Hora, en el formato que deseen y calculada justo antes de la grabación.
- WHERE: ¿Dónde?
 - Comando (parámetro 1), nombre del comando o función que genera el mensaje.
 - Se apreciará la utilidad de este parámetro por ejemplo cuando la función MoverArchivos deba generar mensajes en el log del comando llamador
- WHAT: ¿Qué?
 - Tipo de Mensaje (Parámetro 3) valores posibles:
 - INFO = INFORMATIVO: mensajes explicativos sobre el curso de ejecución del comando. Ej: Inicio de Ejecución
 - WAR = WARNING: mensajes de advertencia pero que no afectan la continuidad de ejecución del comando. Ej: Archivo duplicado
 - ERR = ERROR: mensajes de error Ej: Archivo Inexistente.
 - Valor default: INFO
- WHY: ¿Por qué?
 - Mensaje (Parámetro 2)

El directorio de log está determinado por la variable de ambiente **LOGDIR**

El nombre del archivo de log es igual al nombre del comando y extensión .log

Requerimientos

- Grabar un archivo distinto para cada comando, en el lugar indicado y con el nombre adecuado.
- La escritura de archivos de log debe ser homogénea para todos los comandos
- Cada registro de log debe responder al estándar W5.
- Se debe controlar el crecimiento del archivo de log



- En todo sistema, es importante evitar el crecimiento INDISCRIMINADO de los archivos de Log. Es por ello que esta función debe proveer un mecanismo para controlarlo y evitarlo
- En este mecanismo se debe tener en cuenta la variable de ambiente **LOGSIZE** que representa el tamaño máximo que puede alcanzar un archivo de log en nuestro sistema.
- Este tamaño máximo es un valor de referencia ya que a los efectos prácticos, todo depende del momento en que se realiza el control.
- Lo importante es que SIEMPRE adopte un mecanismo para mantener controlado el tamaño de un log. Puede adoptar cualquier mecanismo, **aclare en Hipótesis y Aclaraciones Globales** cual fue el que adoptó.
 - **Ejemplo sencillo:** cuando un archivo de log supera **LOGSIZE**, el archivo se trunca dejando las últimas 50 líneas
- Cada vez que se hace la reducción del tamaño del archivo, señalar la situación con el mensaje "Log Excedido" en el propio log

Premisas:

- Si el archivo de log no existe, se debe crear. Si existe se le deben agregar los nuevos registros **siempre**

Función *MostrarBitacora*

Opciones y Parámetros

- Parámetro 1 (obligatorio): comando
- Parámetro 2 (opcional): string a buscar
- Otros parámetros u opciones a especificar por el desarrollador

Descripción

Esta función tiene por objeto generar una visualización amigable del contenido del archivo de log asociado al comando pasado como parámetro.

Debe ser una función que se invoca desde línea.

Si se usa con el parámetro dos, debe mostrar todas las coincidencias con ese string

Es deseable poder filtrar la visualización usando algún otro concepto de búsqueda definido por el desarrollador

Debido al alto grado de libertad que se les permite en el desarrollo de este comando, documentarlo detalladamente



Estructuras y Archivos

Maestro de Concesionarios: **MAEDIR/concesionarios.csv**

Separador de campos: ; punto y coma

Contenido: el archivo publicado con el enunciado

Campo	Descripción /Fuente/Valor
Razón Social	N caracteres
Código de Concesionario	Numérico de 4 dígitos

Ejemplos:

AUTOGAONA, S.A.	3741
AUTOMOTORES ESSHOP, S.R.L.	3724
AUTOMOTORES OESTE, S.R.L.	3764

Tabla de Fechas de Adjudicación: **MAEDIR/FechasAdj.csv**

Separador de campos: ; punto y coma

Contenido: el archivo publicado con el enunciado

Campo	Descripción /Fuente/Valor
Fecha	Fecha del Acto de Adjudicación formato: día/mes/año
Lugar	N caracteres

Ejemplos:

14/04/2016	AUTO MAAR, S.R.L.
19/05/2016	AUTO ARG, S.R.L.
16/06/2016	CLAIRE MARCON, S.A.

Archivos de ofertas: **ARRIDIR/<cod_concesionario>_<aniomesdia>.csv**

Separador de campos: ; punto y coma

Archivos de ejemplo publicados junto con el enunciado

Campo	Descripción /Fuente/Valor
Contrato Fusionado	7 caracteres. Los primeros 4 caracteres corresponden al número de grupo, los siguientes 3 corresponden al Numero de orden del suscriptor dentro del grupo
Importe de la Oferta	Importe que se ofrece para licitar

Ejemplos:

7886009	145000
7888036	87000
7888120	84001

Padrón de suscriptores: **MAEDIR/temaK_padron.csv**

Separador de campos: ; punto y coma

Contenido: el archivo publicado con el enunciado



Campo	Descripción /Fuente/Valor
Grupo	4 caracteres, Numero de Grupo
Orden	3 caracteres, Numero de orden del suscriptor dentro del grupo
Nombre del Suscriptor	N caracteres, Apellido y Nombre del suscriptor
Concesionario	4 caracteres, Código del concesionario
Coeficiente	Valor numérico, coeficiente de conversión
Participa?	1 carácter, Marca de participación. Valores posibles: 1 si participa, 2 condicional, blanco no participa
Motivo	2 caracteres, Motivo de la no participación, puede ser blanco
Cuotas de recupero	Puede ser 000000
Cuotas de deuda	Puede ser 00
Fecha primer vencimiento	Fecha, Puede ser 00000000
Primer cuota con deuda	Puede ser 00
Deuda	Importe total de deuda, puede ser 0000000000
Id suscripción	Numero de suscripción

Ejemplos:

7886;009;MARTIN,AGUSTIN ;3500;18752;1;GF;000000;00;00000000;00;0000000000;01661654
7886;012;MATTIOLI,PONS ;2500;18752;1;GF;000000;00;00000000;00;0000000000;01665788
7886;041;MIRRA,LUISA ;3007;18752;1;GF;000000;00;00000000;00;0000000000;01692951
7886;081;NAPAL,LUIS ;3238;18752;1;GF;000000;00;00000000;00;0000000000;01667481

Tabla de Grupos: **MAEDIR/Grupos.csv**

Separador de campos: ; punto y coma

Contenido: el archivo publicado con el enunciado

Campo	Descripción /Fuente/Valor
Nro de Grupo	4 Caracteres. Identifica al Grupo
Estado del Grupo	N caracteres, valores posibles NUEVO, ABIERTO, CERRADO
Cantidad de Cuotas	Numérico. Cantidad total de Cuotas de Plan de Ahorro previo
Cuota Pura	Importe. Representa el Valor de una cuota
Cantidad de cuotas pendientes	Numérico. Nos indica la cantidad de cuotas pendientes del Plan
Cantidad de cuotas para licitación	Numérico. Representa el mínimo de cuotas necesarias para licitar

Ejemplos:

8849	CERRADO	84	1048,00	5	12
8850	ABIERTO	84	1053,20	78	1
8856	ABIERTO	84	1050,00	79	1
8857	ABIERTO	84	1113,60	79	1
8858	NUEVO	84	2045,50	80	12

Archivo de ofertas válidas: **PROCDIR/validas/<fecha_de_adjudicacion>.txt**

Separador de campos: ; punto y coma

Campo	Descripción /Fuente/Valor
Código de Concesionario	Código de concesionario proveniente del nombre del archivo
Fecha del archivo	Fecha del nombre del archivo, formato a elección



Contrato Fusionado	proveniente del archivo de ofertas
Grupo	Primeros 4 caracteres del Contrato
Nro de Orden	Últimos 3 caracteres del Contrato
Importe Ofertado	Importe proveniente del archivo de ofertas
Nombre del Suscriptor	Apellido y nombre del suscriptor, proveniente del padrón de suscriptores
usuario	Login del usuario que graba el registro
fecha	Fecha y hora de grabación del registro, en el formato que se desee

Separador de campos: ; punto y coma

Archivo de ofertas rechazadas:

PROCDIR/rechazadas/<cod_concesionario>.rech

Separador de campos: ; punto y coma

Campo	Descripción /Fuente/Valor
Fuente	Nombre del archivo de input
Motivo	Motivo por el cual se rechaza ESTA oferta
Registro de Oferta	Registro Original COMPLETO
usuario	Login del usuario que graba el registro
fecha	Fecha y hora de grabación del registro rechazado, en el formato que se desee

Archivos de sorteos **PROCDIR/sorteos/<Id_Sorteo>_<fecha_adj>.srt**

Separador de campos: ; punto y coma

Campo	Descripción /Fuente/Valor
Numero de Orden	Valores posibles: 1 .. 168 en forma creciente
Nro de sorteo	Resultado aleatorio obtenido

Archivos de Log: **LOGDIR/<nombre del comando>.log**

Campo	Descripción /Fuente/Valor
Quien	Caracteres Es el login del usuario que graba el registro
Cuando	Fecha y hora Formato a Elección Es la fecha y hora en el momento de grabación del registro.
Donde	Caracteres Nombre del Comando, función o rutina en donde se produce el evento que se registra en el log
Que	Caracteres Lo determina el programador.
Porque	Caracteres Lo determina el programador.

Separador de campos: - guion

Ejemplo: 20150905 19:53:22-Sandra-RecibirOfertas-WAR-No se pudo mover el archivo

En la variable de ambiente LOGSIZE se tiene el tamaño máximo en Kbytes que puede alcanzar un archivo de log

FIN