CARÁTULA

Tema	Grupo	Ayudante	Correcciones			
			Fecha	Hora Inicio	Hora Fin	Resultado
K	5		Entrega			
			Revisión			

INTEGRANTES

	Padrón	Apellido y Nombre	Asistencia a Entrega	Asistencia a Revisión	Evaluación Individual Final
1	95604	Etchanchú, Facundo Matías			
2	95606	Bert, Tomás Agusto			
3	95615	Negri, Guido			
4	95505	Iogha, Octavio			
5	95874	Cura, Martín			
6	95548	Bouvier, Juan Manuel			

Índice del Contenido de la Carpeta

Hipótesis y Aclaraciones Globales

Absolutamente todos los directorios utilizados y pasados como parámetros entre funciones y scripts deben terminar su path con la barra delimitadora de directorios “/”.

PrepararAmbiente

Es el único script que tiene permitido el acceso al archivo de configuración del sistema.

RecibirOfertas

En cada ciclo se crean archivos auxiliares temporales los cuales son eliminados inmediatamente tras su uso, por lo tanto persisten únicamente por décimas de segundo.
Los archivos maestros consultados no son necesariamente validados en su existencia, tipo, nombre, etc., ya que de esto se ocupa **PrepararAmbiente**. Al consultar a la cátedra, confirmaron que los maestros se pueden suponer confiables.
Al realizar la consulta de la última fecha de un acto de adjudicación en el archivo correspondiente se considera que estos se encuentran en orden cronológico, lo cual es cierto para el maestro provisto pero no es una precondición del enunciado.

ProcesasOfertas

Los archivos de ofertas tienen un nombre válido, ya sea la fecha y el código del concesionario que componen el nombre, pues este proceso es llamado únicamente por el script de **RecibirOfertas** y es este el encargado de dichas validaciones.
Si alguna línea de al ǵ un .csv del *MAEDIR* no cumple con la estructura detallada en el enunciado, se rechaza la oferta asociada a esa línea (sea suscriptor o grupo), incluso en los casos en los que el error no podría afectaría el procesamiento de la oferta.
Si la oferta que se está procesando ya fue validada se comparan el importe de la oferta anterior y el de la nueva. Si el nuevo importe es mayor o igual al anterior entonces se reemplaza la oferta en el archivo de válidas. En caso contrario, se rechaza la oferta procesada y se arroja un mensaje indicando que ya existe una oferta análoga con importe más alto.
La fecha de adjudicación puede ser la misma que la fecha del archivo de oferta.
No se asume que la **Tabla de Fechas de adjudicación** esté ordenada, por lo tanto se comparan todas las fechas.
Siempre habrá una próxima fecha de adjudicación, pues filtrar las que no tienen es el trabajo del script **RecibirOfertas**.
Se asume que los valores para calcular máximo y mínimo son números. Además, en caso de ser un número decimal se utiliza coma decimal (“,”) en lugar de punto. Si se encuentra un número con punto, el mismo será considerado inválido y descartado del registro.

GenerarSorteo

Si no se encuentra el archivo **FechasAdj.csv** se registra el error correspondiente en la bitácora y termina su ejecución.
El ID del sorteo inicialmente es 1, si encuentra uno ya generado para la misma fecha incrementa el ID en una unidad.

DeterminarGanadores

Todos los parámetros se reciben por <STDIN>, a excepción del argumento -g o -a, de grabación o ayuda respectivamente.
De no encontrarse algún archivo en particular (sea el de sorteo, adjudicación, o cualquiera de los inputs especificados en el documento) se reinicia la ejecución del script, de esa manera tolerando errores humanos, y a su vez cumpliendo con el requisito de poder tomar N consultas.
Los resultados de las consultas se imprimen por <STDOUT> y, opcionalmente, a archivos de texto.
Se tomó como nomenclatura para indicar rango de grupos, utilizar como separador el carácter “-”, como se indica en el script.
Se ha tomado como ganador del sorteo, a aquel que obtenga el número más bajo de sorteo dentro del grupo.
El ganador de la licitación es, aquella persona que tenga la oferta válida de mayor importe dentro del grupo, y que no coincide con la persona ganadora del sorteo.
La única excepción en la cual puedan coincidir ganador de licitación y sorteo, es cuando el grupo tiene una sola oferta válida, en tal caso esa persona resultaría ganadora en ambos casos.
Por enunciado, los archivos a grabar de las consultas por sorteo y licitación, podrían llegar a ser iguales, pisándose los mismos de esa manera. Se optó por agregar un identificador al nombre del archivo de la licitación, en este caso el sufijo “_licitacion”.
No se graba ningún error en el archivo log.
Se supone que los archivos de INPUT son correctos, por lo cual no se realizan algunas verificaciones.

MoverArchivos

Implementado en Shell.
Se emplea un archivo auxiliar por cada directorio para llevar registro de la cantidad de duplicados y su numeración. De esta forma se evita el acceso al archivo de configuración.
Si no se especifica el comando llamador de esta función complementaria entonces el resultado del script se imprime por *STDOUT*.

GrabarBitacora

Implementado en Shell.
La función no registra en la bitácora mensajes vacíos.

Si no se especifica un tipo de mensaje, por defecto se registra el mensaje como INFO (informativo).
Cuando se supere el *LOGSIZE* máximo el archivo de bitácora correspondiente se limpia y se ingresan sus últimas cincuenta (50) líneas, además de la línea informativa al final aclarando que el log fue excedido.

MostrarBitacora

Implementado en Perl.
El script busca coincidencias de acuerdo a expresiones regulares.

LanzarProceso

Implementado en Shell.
El chequeo que verifica si el ambiente fue correctamente inicializado o no previo a lanzar un proceso se centraliza en este script. Así, se evita realizar el chequeo en cada script invocador.
El chequeo de ambiente consiste simplemente en verificar si la variable de entorno *$AMBIENTE_INICIALIZADO* tiene valor booleano true o false.
Todos los procesos lanzados se ejecutan en *background* por defecto, salvo que se settee el flag correspondiente a *foreground*.
Si el script es invocado desde el mismo shell (por ejemplo, como *source*) se considera que se lo está llamando desde otro script y por lo tanto no se imprime por salida standard la información de su ejecución; en cambio, al invocarse normalmente en un shell hijo (por ejemplo, anteponiendo el comando *bash*) se considera que se lo está llamando desde la línea de comandos y entonces sí se muestra el resultado por salida standard.

DetenerProceso

Implementado en Shell.
Detener un proceso implica matarlo, no dejarlo en pausa.
Para chequear si el proceso ya existe, se busca el PID de la primera palabra del nombre del proceso pasado como parámetro. Además, esa palabra se recorta a 15 caracteres como máximo ya que es así como maneja internamente los procesos Unix. En caso de encontrarse más de un PID por haberse encontrado varios procesos iguales pero con distintos parámetros, se mata el primer proceso de la lista.
De acuerdo a los requerimientos, toda la información de ejecución del script se imprime por salida standard además de registrarse en bitácoras, sin excepciones.

Problemas relevantes

Uno de los inconvenientes que enfrentó el grupo durante el desarrollo del sistema fue más debido a una indecisión que un obstáculo propio de implementación.
Se discutió cómo deberían ser lanzados los procesos, todos ellos puesto que la ejecución de procesos está centralizada en un único comando. Debido a que **RecibirOfertas** es un daemon que corre constantemente en *background*, se optó finalmente por correr todos los procesos de forma análoga, por lo que toda la lógica de invocación al comando correspondiente realizada en **LanzarProceso** no debería interrumpir el normal flujo del sistema.
Sin embargo, esto reveló otra cuestión quizás no tan importante en la teoría pero sí muy influyente a la hora de implementar el comando: ¿cómo debería ser invocado **LanzarProceso**? El problema radica en que no se encontró una forma determinística de chequear si el comando fue invocado desde la línea de comandos o desde otro script y un requisito del enunciado implica discriminar ambos casos para definir si debe imprimirse la información relacionada a su ejecución por salida estándar o registrarla en bitácoras, respectivamente.
Como “parche” o solución alternativa, se llegó a la conclusión de que sería más “intuitivo” llamar a la función dentro del mismo shell en el que se está trabajando, por lo tanto se invoca el script como *source*, es decir anteponiendo el alias “.”. Por otro lado, cuando se llama al script por línea de comandos parece adecuado hacerlo anteponiendo el término *bash*, esto es creando un subshell hijo. De esta forma, el script evaluará cuál fue su comando *caller* para determinar si debe imprimir por pantalla o registrar en bitácora su output.

Otro inconveniente devino del llamado a **PrepararAmbiente**. Se supone que el manejo de procesos está centralizado en **LanzarProceso** y **DetenerProceso**, no obstante para lanzar un proceso con el primer comando es obligatorio que el ambiente ya esté definido; nótese cierto “vacío legal”. En este caso particular se decidió hacer una excepción y llamar al comando **PrepararAmbiente** directamente, sin utilizar como ejecutor al script de **LanzarProceso**.

Archivo README

CIPAK | Sistemas Operativos (75.08) | 1er Cuatrimestre 2016 | FIUBA

Descripción del sistema

Instalar

Para instalar el sistema **CIPAK** en su computadora, coloque un dispositivo USB con el archivo CIPAK-G5.tgz, luego elija un directorio donde descomprimirlo, realice la descompresión del mismo (puede hacerlo mediante clic derecho, y la opción *Extract here*), y corra el instalador dentro de la carpeta, mediante la siguiente línea de comando:
.../CIPAK-G5$. installer.sh
El instalador debe estar en el mismo directorio que el fichero comprimido source.tar.gz para poder realizar la instalación con éxito.

Desinstalar

Para desinstalar el sistema, vaya a la carpeta de binarios (por default, *Grupo05/binarios/*) y corra el desinstalador **uninstaller.sh**, lo cual puede hacer mediante: .../Grupo05/binarios$. uninstaller.sh

Preparar el ambiente

Una vez instalado **CIPAK** ejecute en una terminal en la dirección *Grupo05/binarios* el comando .../Grupo05/binarios$. PrepararAmbiente.sh Mediante esto usted tendrá todo preparado para poder ejecutar los programas del sistema.

Recibir ofertas

Una vez iniciado el ambiente, puede en cualquier momento (el paso anterior ofrece hacerlo automáticamente) iniciar el *daemon* receptor mediante .../Grupo05/binarios$ bash LanzarProceso.sh RecibirOfertas.sh con lo cual se iniciará en un segundo plano el proceso que medie la recepción de archivos de oferta. Para frenarlo, utilice el mismo formato pero con el comando DetenerProceso.sh.

Procesar ofertas

ProcesarOfertas es llamado por RecibirOfertas cuando hay novedades. Una vez finalizado se pueden ver los resultados en los outputs correspondientes referenciados en la documentación.

Generar sorteo

Una vez que el ambiente esté listo podrá generar el sorteo mediante .../Grupo05/binarios$ bash LanzarProceso.sh "bash GenerarSorteo.sh" Esto genera los archivos necesarios para poder determinar los ganadores.

Determinar ganadores

Habiendo preparado el ambiente, y posteriormente generado el sorteo correspondiente Posicionarse en .../Grupo05/binarios/ y ejecutar el comando mediante la sentencia $ ./DeterminarGanadores.pl. Se accederá entonces al menu correspondiente a las consultas. Dentro de las opciones del comando, usted puede utilizar:
-g para grabar las consultas realizadas en un archivo de texto. $ ./DeterminarGanadores.pl -g
-a para acceder a la ayuda correspondiente al comando. $ ./DeterminarGanadores.pl -a

Lanzar y detener procesos

Toda la lógica de ejecución de procesos está centralizada en las funciones **LanzarProceso** y **DetenerProceso**. Los procesos son ejecutados en background, demonizados. La única excepción en la que no DEBE llamarse a **LanzarProceso** para ejecutar otro proceso es en el caso de la función de **PrepararAmbiente** pues, por requerimiento, un proceso no puede ser lanzado sin estar inicializado el ambiente.

Modo de uso de scripts

installer.sh

Si recibió el paquete **CIPAK_G5.tgz**, el primer paso es descomprimirlo. Esto se puede hacer mediante clic derecho y eligiendo la opción de descompresión en la carpeta actual, o con: tar -xzf CIPAK_G5.tgz
Tras esto, dispondrá de una carpeta del mismo nombre con todo los archivos que necesita la instalación, junto con esta documentación.
El instalador no requiere ningún parámetro para su funcionamiento. Para ejecutarlo, se recomienda que se utilice el siguiente comando, con la notación de . para que la instalación se realice en el mismo proceso que la terminal, y con el *working directory* posicionado en la carpeta del mismo, de forma que pueda funcionar totalmente: . installer.sh
Opcionalmente, se puede utilizar el flag -d para definir todas las variables de ambiente diferentes de aquellas por default: . installer.sh -d

uninstaller.sh

El desinstalador no utiliza ningún parámetro. Para ejecutarlo, se recomienda que se utilice el siguiente comando, con la notación de . para que la desinstalación se realice en el mismo proceso que la terminal y con el *working directory* posicionado en la carpeta del script, de forma que pueda garantizarse su total funcionamiento: . uninstaller.sh

PrepararAmbiente.sh

No requiere ningún parámetro.
El script necesita imperiosamente de los siguientes archivos para lograr su cometido:

- Archivo de Configuración: *CONFDIR/CIPAK.cnf*
- Scripts ejecutables: *BINDIR/{script}.sh/.pl*
- Archivos maestros: *MAEDIR/{maestro}.csv*
- Directorio de resguardo: A definir por el instalador.
-

El script define las siguientes variables de ambiente:

- Por enunciado:
    - GRUPO
    - MAEDIR
    - ARRIDIR
    - OKDIR
    - PROCDIR
    - INFODIR
    - LOGDIR
    - NOKDIR
    - LOGSIZE
    - SLEEPTIME
    - PATH
- Agregadas:
    - REGSDIR: Donde se encuentra el repositorio de resguardo.
    - AMBIENTE_INICIALIZADO: flag para informar que el ambiente fue inicializado correctamente.
    -

El script devuelve alguno de los siguientes valores:

- **0**: Éxito. El usuario decide no continuar la ejecución del sistema con **RecibirOfertas**.
- **1**: Error. No se inicializó el sistema (falta el archivo de configuración o ya se encuentra inicializado).
- **2**: Error. No se poseen todos los scripts obligatorios.
- **3**: Error. No se poseen todos los archivos maestros obligatorios.
- **4**: Error. No se poseen todos los permisos necesarios en los scripts o los archivos maestros.
-

RecibirOfertas.sh

No requiere ningún parámetro.
Se lo ejecuta opcionalmente tras el proceso de **PrepararAmbiente** o bien independientemente.
Si el ambiente no había sido inicializado, de cualquier manera, el programa no correrá.
El script necesita imperiosamente de los siguientes archivos para lograr su cometido:

- Archivos de oferta: *ARRIDIR/{cod_concesionario}_{aniomesdia}.csv*
- Tabla de fechas de adjudicación: *MAEDIR/FechasAdj.csv*

- Registro de concesionarios: *MAEDIR/concesionarios.csv*
-

El resultado de un ciclo del programa es la separación de los archivos de oferta de *input* en aquellos válidos e inválidos según una serie de criterios, con lo que el *output* será:

- Archivos de oferta válidos: *OKDIR/{cod_concesionario}_{aniomesdia}.csv*
- Archivos de oferta inválidos: *NOKDIR/{cod_concesionario}_{aniomesdia}.csv*
-

ProcesarOfertas.sh

No requiere ningún parámetro.
El script necesita imperiosamente de los siguientes archivos para lograr su cometido:

- Archivo 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*
-

Desde estos input genera como resultado-output:

- Archivo de ofertas válidas: *PROCDIR/validas/{fecha_de_adjudicacion}.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}*
-

GenerarSorteo.sh

No recibe parámetros.
El script necesita imperiosamente del siguiente archivo para lograr su cometido:

- Tabla de Fechas de adjudicación: *MAEDIR/FechasAdj.csv*
-

Desde este input genera como resultado-output:

- Archivos de sorteos: *PROCDIR/sorteos/{sorteoId}{fecha_de_adjudicacion}.csv*, donde se indica para cada una de las 168 órdenes el número aleatorio que se le asignó.
-

En caso de realizarse un flow completo del script, es decir que llegue al final de su ejecución sin ningún inconveniente, generará un archivo SRT para la fecha más próxima alojada en la Tabla de Fechas de adjudicación.

DeterminarGanadores.pl

El script acepta 2 flags diferentes:
1. -a: Ayuda. Se imprime en pantalla la ayuda del comando, esto es qué significa cada opción y cómo navegar a través del script.
2. -g: Modo grabar. Es la opción para grabar los resultados de las consultas en archivos, cuyos nombres son representativos de las consultas realizadas. Por salida estándar se avisará que las consultas se están grabando en disco.

En caso de llamarse sin argumentos al comando, se realizan consultas de tipo provisorias, es decir sólo se ven en pantalla y no persisten en el disco.
El script necesita imperiosamente de los siguientes archivos para funcionar correctamente:

- Padrón de Suscriptores: *MAEDIR/temaK_padron.csv*
- Tabla de Grupos: *MAEDIR/grupos.csv*
- Archivo de ofertas válidas: *PROCDIR/validas/{fecha_de_adjudicacion}.txt*
- Archivos de sorteos: *PROCDIR/sorteos/{sorteoId}_{fecha_de_adjudicacion}.srt*
-

Si se corre en modo grabar (-g), desde estos input se genera como resultado-output:

- Resultado general del sorteo: *INFODIR/{sorteoId}_{fecha_de_adjudicacion}.txt*
- Ganadores por Sorteo: *INFODIR/{sorteoId}Grdxxxx-Grhyyyy{fecha_de_adjudicacion}*
- Ganadores por Licitación: *INFODIR/{sorteoId}Grdxxxx-Grhyyyy{fecha_de_adjudicacion}*
- Resultados por grupo: *INFODIR/{sorteoId}Grupoxxxx{fecha_de_adjudicacion}*
-

MoverArchivos.sh

El script acepta hasta 3 argumentos distintos:
1. **ORIGEN** *(Requerido)*. Nombre del archivo a mover. 2. **DESTINO** *(Requerido)*. Nombre del directorio adonde se quiere mover el archivo origen.
3. **COMANDO** *(Opcional)*. Indica el nombre del comando invocador, utilizado dentro del script para loggear en la bitácora correspondiente la información pertinente. Si no se especifica, la información de ejecución se imprime por *STDOUT*.
El script devuelve alguno de los siguientes valores:

- **0**: Éxito. El archivo fue movida del origen al destino sin necesidad de realizar duplicados o colocarlo en una carpeta auxiliar.
- **1**: Éxito. El archivo pudo ser movido exitosamente pero debió guardarse como copia en la carpeta auxiliar *dpl/*.
- **2**: Error. El destino corresponde al mismo directorio donde se halla actualmente el archivo origen y por lo tanto no se puede mover el archivo.
- **3**: Error. El archivo origen no se encuentra en el directorio específicado.
- **4**: Error. El directorio destino no existe.
-

GrabarBitacora.sh

El script acepta hasta 3 argumentos distintos:
1. **COMANDO** *(Requerido)*. Nombre del comando invocador del script. El archivo de bitácora donde se plasmarán los registros se llamará *{COMANDO}.log*.
2. **MENSAJE** *(Requerido)*. El mensaje a registrar en la bitácora.
3. **TIPO DE MENSAJE** *(Opcional)*. Indica la categoría del mensaje a registrar en la bitácora. Puede ser INFO, WARNING o ERROR. Si no se define este parámetro, el valor por defecto es INFO.
Se utilizan las siguientes variables de ambiente:

- *LOGDIR*: Directorio donde se crearán los archivos de log.
- *USER*: Usuario actual que escribe el registro en la bitácora.

- *LOGSIZE*: Tamaño máximo de los archivos de log en KBytes.
-

MostrarBitacora.pl

El script acepta hasta 3 argumentos distintos:
1. **BITACORA** *(Requerido)*. Nombre del archivo bitácora sobre el cual se va a realizar la consulta. Debe especificarse ruta relativa y extensión del archivo.
2. **QUERY** *(Opcional)*. Puede ser tanto una cadena de texto normal como una expresión regular. Funciona como filtro puesto que se mostrarán sólo las líneas de la bitácora correspondiente que matcheen con la query ingresada. En caso de no indicarse este parámetro, se mostrarán todas las líneas de la bitácora.
3. **ARCHIVO DE SALIDA** *(Opcional)*. Es el nombre y extensión del archivo en el cual se imprimirán las líneas de la bitácora matcheadas por la query. En caso de no indicarse, obviamente se imprimirán por *STDOUT* por defecto. El archivo tendrá ubicación relativa al directorio donde se está ejecutando el script. Si el archivo de salida especificado no existe, lo crea; si ya existe, las líneas se insertan al final del mismo, en orden.
El script devuelve alguno de los siguientes valores:

- **0**: Éxito. El archivo bitácora fue abierto correctamente. No se distingue si la consulta fue *matcheada* o no en el valor de retorno.
- **1**: Error. No se pudo abrir el archivo de bitácora sobre el que se va a realizar la consulta.
- **2**: Error. No se pudo abrir o crear el archivo de salida del script.
-

LanzarProceso.sh

Es necesario que la variable de ambiente AMBIENTE_INICIALIZADO esté setteada como verdadera (1) para poder lanzar cualquier proceso.
El script acepta hasta 2 argumentos distintos:
1. **PROCESO** *(Requerido)*. Nombre completo del proceso a lanzar, incluidos todos los parámetros que este reciba. Debemos imaginar este argumento como un comando cualquiera que queramos ejecutar en una línea de shell en la terminal.
2. **COMANDO** *(Opcional)*. Indica el comando desde el cual se invoca al script. En caso de no pasarse este argumento, en caso de querer registrarse algo en la bitácora no se podrá puesto que no estará especificado el comando correspondiente a la misma.
El comportamiento por defecto de la función es lanzar los procesos en *background*. No obstante, se puede cambiar este comportamiento para que el proceso sea lanzado en primer plano setteando el *flag* opcional -f o --foreground.
Se recomienda invocar este comando como bash LanzarProceso.sh "{PROCESO}" cuando se lo quiera llamar desde la línea de comandos. Por el contrario, se sugiere hacerlo *source* (alias .) en aquellos casos en los que se lo ejecute desde otro script: . LanzarProceso.sh "{PROCESO}". El porqué de esta diferenciación tiene que ver con cómo deberá manejar el script el registro de errores e información.
El script devuelve alguno de los siguientes valores:

- **0**: Éxito. El proceso especificado pudo ser lanzado correctamente.
- **1**: Error. El proceso a lanzar ya está en ejecución.

- **2**: Error. Ante cualquier otro error al querer lanzar el proceso en background.
- **3**: Error. El ambiente no fue inicializado.
- **4**: Error. La función no recibió parámetros, es decir que no se especificó el proceso a lanzar.
-

DetenerProceso.sh

El script acepta hasta 3 argumentos distintos:
1. **PROCESO** *(Requerido)*. Nombre completo del proceso a detener.
2. **COMANDO** *(Requerido)*. Nombre del comando desde donde se invoca esta función. Este argumento sirve para registrar en la bitácora correspondiente a dicho comando todo lo referente a la ejecución del script.
3. **PID** *(Opcional)*. Indica el PID del proceso a detener. En caso de no especificarse, se busca el PID entre los procesos en ejecución de acuerdo al nombre del proceso. Cabe aclarar que, por cómo se manejan los procesos en el sistema y en UNIX en general, el nombre del proceso puede matchear con varios procesos con PIDs distintos; en ese caso, se selecciona el primer PID de la lista.
El script devuelve alguno de los siguientes valores:

- **0**: Éxito. El proceso se estaba ejecutando y fue detenido con éxito. En realidad, el valor de retorno que se devuelve es el correspondiente al comando kill {PID}.
- **1**: Error. El proceso a detener no estaba ejecutándose.
- **2**: Error. La función no recibió parámetros, es decir que no se especificó el proceso a detener.

Listado de Nuevas Funciones y/o Comandos Auxiliares

compiler.sh

Este comando crea la carpeta CIPAK_G5 dentro del directorio actual. En esa carpeta se colocará el archivo comprimido source.tar.gz requerido por el instalador, el instalador installer.sh y además un Readme explicando a grandes rasgos el uso del sistema.

installer.sh

El propósito de este comando es instalar el sistema CIPAK_G5. Se encarga de organizar los archivos, registrar las variables de ambiente y grabar la configuración.
Para poder correrlo, requiere del archivo acompañante source.tar.gz, desde donde se produce la instalación, en la misma carpeta. Al iniciar la instalación, pedirá valores para las distintas variables de ambiente; en cualquier caso se recomienda dejar estos campos vacíos, y el instalador se ocupará de tomar los valores por default (que se mostrarán en pantalla).
El resultado esperado de la instalación es el sistema correctamente instalado y la terminal donde se la corrió lista para su uso en el directorio principal del mismo.
El comando debe ser invocado manualmente.

uninstaller.sh

El propósito de este comando es desinstalar el sistema CIPAK_G5. Se encarga de recrear los archivos de instalación, limpiar la instalación y la configuración.
Correrlo es sumamente sencillo, y solo requiere que la copia de resguardo no haya sido adulterada por el usuario. Tras una pequeña verificación de que está seguro de la acción, el sistema será eliminado sin más preámbulo.
El resultado esperado de la desinstalación es el sistema correctamente desinstalado y la persistencia de los archivos del paquete inicial descargado.
ADVERTENCIA: La desinstalación es un proceso definitivo sobre los archivos guardados en las carpetas. Cualquier archivo subsistente en las subcarpetas del sistema, sean de novedades o producto de su procesamiento, serán eliminados si no se crea una copia de aquellos que no desea perder.
No hay forma de revertir lo hecho por este programa sobre los archivos *input*.

funcionesDeChequeo.sh

El propósito del archivo es alojar en él todas las funciones de chequeos de los archivos maestros (como ejemplo la validación de una fecha en FechasAdj.csv). De esta manera podemos reutilizar en todos los script las mismas funciones.

Listado de DATOS

#Registros | NombreArchivo

```
9851 temaK_padron.csv
880 grupos.csv
104 concesionarios.csv
11 FechasAdj.csv
312 Novedades/3402_20160420.csv
194 Novedades/3776_20160506.csv
192 Novedades/3500_20160420.csv
183 Novedades/3748_20160420.csv
157 Novedades/3759_20160503.csv
155 Novedades/3760_20160504.csv
148 Novedades/3783_20160510.csv
147 Novedades/3741_20160426.csv
134 Novedades/3520_20160422.csv
117 Novedades/3771_20160506.csv
116 Novedades/3510_20160420.csv
104 Novedades/3778_20160502.csv
102 Novedades/3729_20160424.csv
94 Novedades/3779_20160509.csv
93 Novedades/3303_20160420.csv
91 Novedades/3505_20160421.csv
84 Novedades/3765_20160510.csv
81 Novedades/4501_20160510.csv
78 Novedades/3766_20160510.csv
76 Novedades/3724_20160420.csv
64 Novedades/3764_20160510.csv
63 Novedades/3735_20160425.csv
55 Novedades/3761_20160504.csv
53 Novedades/3785_20160510.csv
43 Novedades/3769_20160502.csv
42 Novedades/4503_20160510.csv
40 Novedades/3753_20160430.csv
38 Novedades/3758_20160503.csv
38 Novedades/3754_20160502.csv
36 Novedades/3756_20160503.csv
36 Novedades/3744_20160420.csv
35 Novedades/3732_20160420.csv
30 Novedades/3750_20160420.csv
29 Novedades/3770_20160506.csv
29 Novedades/3768_20160510.csv
25 Novedades/3762_20160504.csv
25 Novedades/3739_20160420.csv
25 Novedades/3722_20160423.csv
24 Novedades/3757_20160503.csv
19 Novedades/3718_20160420.csv
18 Novedades/3775_20160506.csv
17 Novedades/3773_20160506.csv
16 Novedades/3763_20160502.csv
15 Novedades/3746_20160420.csv
10 Novedades/3745_20160427.csv
4 Novedades/3784_20160510.csv
3 Novedades/3782_20160510.csv
3 Novedades/3780_20160507.csv
```

Listado de Nuevos Archivos

secuencia.aux

En realidad, se trata de varios archivos homónimos. Cada uno corresponde a un directorio diferente y se crearán tantos como sean necesarios. Son archivos permanentes.
Este archivo posee un sólo registro que es un número entero, iniciando en 1. Básicamente, funciona como un contador incremental y es complementario al comando **MoverArchivos**. Cuando el comando encuentra que el archivo ya existe en el origen entonces crea la carpeta dpl/ en ese directorio y lo coloca allí; si esta carpeta ya existía y, por lo tanto, ya tenía un archivo en ella, el comando debe crear este archivo en dicha carpeta para llevar cuenta del número de copias efectuadas y así poder renombrar cada vez el archivo movido con el sufijo indicado.

Hoja de ruta de prueba “camino feliz”

- Instale el tp e Imprima el contenido del archivo de configuración

```
$ cat CONF/CIPAK.cnf
LOGDIR=/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/bitacoras/=facu=01/05/2016 12:41:20
LOGSIZE=500000=facu=01/05/2016 12:41:20
GRUPO=/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/=facu=01/05/2016 12:41:20
BINDIR=/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/binarios/=facu=01/05/2016 12:41:20
MAEDIR=/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/maestros/=facu=01/05/2016 12:41:20
ARRIDIR=/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/arribados/=facu=01/05/2016 12:41:20
OKDIR=/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/aceptados/=facu=01/05/2016 12:41:20
PROCDIR=/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/procesados/=facu=01/05/2016 12:41:20
INFODIR=/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/informes/=facu=01/05/2016 12:41:20
RESGDIR=/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/source/=facu=01/05/2016 12:41:20
NOKDIR=/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/rechazados/=facu=01/05/2016 12:41:20
SLEEPTIME=10=facu=01/05/2016 12:41:20
```

- Imprima las primeras 10 líneas de contenido de los archivos maestros

```
$ head -n 10 MAEDIR/concesionarios.csv
AG - SERVICIOS AUTOMOTORES, S.R.L.;3500
ALEXANDER NIGEL, S.A.;2500
ALTO LOITTES, S.R.L.;3007
ALTO SCALLA, S.R.L.;3238
ARTEGRAN, S.R.L.;1554
AUTO ARG, S.R.L.;1055
AUTO ESTADO, S.R.L.;4503
AUTO MAAR, S.R.L.;1051
AUTO MARKET, S.A.;2720
AUTO MARSELLA, S.A.;1164

$ head -n 10 MAEDIR/FechasAdj.csv
14/01/2016;TAXI MARTI, S.R.L.
18/02/2016;TEC MOTORS, S.A.I.C.
17/03/2016;TECNICA MOTOR, S.A.I.C.
14/04/2016;AUTO MAAR, S.R.L.
19/05/2016;AUTO ARG, S.R.L.
16/06/2016;CLAIRE MARCON, S.A.
14/07/2016;CITROMEL, S.R.L.
18/08/2016;SCHUMM, S.A.
15/09/2016;RECORD AUTO, S.A.
13/10/2016;CHEVRON, S.A.

$ head -n 10 MAEDIR/grupos.csv
```

```
7886;ABIERTO;84;1875,2;77;13
7888;ABIERTO;84;1875,2;77;13
7903;ABIERTO;84;1875,2;51;1
7890;ABIERTO;84;3004,3;63;13
7891;ABIERTO;84;1714,7;83;1
7915;ABIERTO;84;1217,5;14;1
7893;ABIERTO;84;1875,2;77;13
7894;ABIERTO;84;1875,2;62;13
7895;ABIERTO;84;1221,5;45;13
7896;ABIERTO;84;1875,2;27;13
```

```
$ head -n 10 MAEDIR/temaK_padron.csv
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
7886;084;CABEZAS,CAMILA  ;1554;18752;1;GT;000000;00;00000000;00;0000000000;01672074
7886;146;CEJAS,MARIA ;1055;18752;1;GM;000000;00;00000000;00;0000000000;01659783
7888;010;GIMENEZ,MIRTA ;1055;18752;1;GM;000000;00;00000000;00;0000000000;01670891
7888;018;ITURROSPE,HORACIO ;1051;18752;1;GT;000000;00;00000000;00;0000000000;01664988
7888;036;LLOVERAS,DANIEL ;3500;18752;1;GT;000000;00;00000000;00;0000000000;01661696
7888;120;LUJAN,MARIA ;3500;18752;1;GF;000000;00;00000000;00;0000000000;01694554
```

- Ejecute PrepararAmbiente y permita que el demonio arranque. Imprima el log de PrepararAmbiente

```
$ cat LOGDIR/PrepararAmbiente.log
facu 01/05/2016 12:41:27 PrepararAmbiente [INFO]: Nombre de variable: LOGDIR - Valor:
/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/bitacoras/
facu 01/05/2016 12:41:27 PrepararAmbiente [INFO]: Nombre de variable: LOGSIZE - Valor:
500000
facu 01/05/2016 12:41:27 PrepararAmbiente [INFO]: Nombre de variable: GRUPO - Valor:
/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/
facu 01/05/2016 12:41:27 PrepararAmbiente [INFO]: Nombre de variable: BINDIR - Valor:
/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/binarios/
facu 01/05/2016 12:41:27 PrepararAmbiente [INFO]: Nombre de variable: MAEDIR - Valor:
/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/maestros/
facu 01/05/2016 12:41:27 PrepararAmbiente [INFO]: Nombre de variable: ARRIDIR - Valor:
/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/arribados/
facu 01/05/2016 12:41:27 PrepararAmbiente [INFO]: Nombre de variable: OKDIR - Valor:
/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/aceptados/
facu 01/05/2016 12:41:27 PrepararAmbiente [INFO]: Nombre de variable: PROCDIR - Valor:
/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/procesados/
```

```
facu 01/05/2016 12:41:27 PrepararAmbiente [INFO]: Nombre de variable: INFODIR - Valor:
/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/informes/
facu 01/05/2016 12:41:27 PrepararAmbiente [INFO]: Nombre de variable: RESGDIR - Valor:
/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/source/
facu 01/05/2016 12:41:27 PrepararAmbiente [INFO]: Nombre de variable: NOKDIR - Valor:
/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/rechazados/
facu 01/05/2016 12:41:27 PrepararAmbiente [INFO]: Nombre de variable: SLEEPTIME -
Valor: 10
facu 01/05/2016 12:41:27 PrepararAmbiente [INFO]: Nombre de variable:
AMBIENTE_INICIALIZADO - Valor: 1
facu 01/05/2016 12:41:27 PrepararAmbiente [INFO]: Nombre de variable: PATH - Valor:
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/game
s:/snap/bin:/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/binarios/
facu 01/05/2016 12:41:27 PrepararAmbiente [INFO]: Estado del Sistema: INICIALIZADO
CORRECTAMENTE.
facu 01/05/2016 12:41:28 PrepararAmbiente [INFO]: El proceso
"/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/binarios/RecibirOfertas.sh" ha sido
lanzado exitosamente.
facu 01/05/2016 12:41:28 PrepararAmbiente [INFO]: El comando RecibirOferta fue
activado.
facu 01/05/2016 12:41:28 PrepararAmbiente [INFO]: Finaliza la ejecucion de
PrepararAmbiente.
facu 01/05/2016 12:41:29 PrepararAmbiente [INFO]: Finaliza la ejecucion 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.

```
$ cat LOGDIR/RecibirOfertas.log
facu 01/05/2016 15:44:19 RecibirOfertas [INFO]: ciclo nro. 1
facu 01/05/2016 15:44:19 RecibirOfertas [INFO]: El archivo
"/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/arribados/3303_20160420.csv" se movió
exitosamente al destino "/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/aceptados/".
facu 01/05/2016 15:44:19 RecibirOfertas [INFO]: Archivo
/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/arribados/3303_20160420.csv aceptado y
movido a /home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/aceptados/3303_20160420.csv
facu 01/05/2016 15:44:19 RecibirOfertas [INFO]: El archivo
"/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/arribados/3402_20160420.csv" se movió
exitosamente al destino "/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/aceptados/".
facu 01/05/2016 15:44:19 RecibirOfertas [INFO]: Archivo
/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/arribados/3402_20160420.csv aceptado y
movido a /home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/aceptados/3402_20160420.csv
```

```
facu 01/05/2016 15:44:19 RecibirOfertas [INFO]: El proceso
"/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/binarios/ProcesarOfertas.sh" ha sido
lanzado exitosamente.
facu 01/05/2016 15:44:19 RecibirOfertas [INFO]: ProcesarOfertas corriendo bajo el no.:
7357
facu 01/05/2016 15:44:29 RecibirOfertas [INFO]: ciclo nro. 2
facu 01/05/2016 15:44:39 RecibirOfertas [INFO]: ciclo nro. 3
facu 01/05/2016 15:44:49 RecibirOfertas [INFO]: ciclo nro. 4
facu 01/05/2016 15:44:59 RecibirOfertas [INFO]: ciclo nro. 5
facu 01/05/2016 15:45:09 RecibirOfertas [INFO]: ciclo nro. 6
facu 01/05/2016 15:45:19 RecibirOfertas [INFO]: ciclo nro. 7
facu 01/05/2016 15:45:29 RecibirOfertas [INFO]: ciclo nro. 8
facu 01/05/2016 15:45:39 RecibirOfertas [INFO]: ciclo nro. 9
facu 01/05/2016 15:45:49 RecibirOfertas [INFO]: ciclo nro. 10
facu 01/05/2016 15:45:59 RecibirOfertas [INFO]: ciclo nro. 11
facu 01/05/2016 15:46:09 RecibirOfertas [INFO]: ciclo nro. 12
facu 01/05/2016 15:46:19 RecibirOfertas [INFO]: ciclo nro. 13
facu 01/05/2016 15:46:29 RecibirOfertas [INFO]: ciclo nro. 14
facu 01/05/2016 15:46:39 RecibirOfertas [INFO]: ciclo nro. 15
facu 01/05/2016 15:46:49 RecibirOfertas [INFO]: ciclo nro. 16
facu 01/05/2016 15:46:59 RecibirOfertas [INFO]: ciclo nro. 17
facu 01/05/2016 15:47:09 RecibirOfertas [INFO]: ciclo nro. 18
facu 01/05/2016 15:47:19 RecibirOfertas [INFO]: ciclo nro. 19
facu 01/05/2016 15:47:29 RecibirOfertas [INFO]: ciclo nro. 20
facu 01/05/2016 15:47:40 RecibirOfertas [INFO]: ciclo nro. 21
facu 01/05/2016 15:47:50 RecibirOfertas [INFO]: ciclo nro. 22
facu 01/05/2016 15:48:00 RecibirOfertas [INFO]: ciclo nro. 23
facu 01/05/2016 15:48:10 RecibirOfertas [INFO]: ciclo nro. 24
facu 01/05/2016 15:48:20 RecibirOfertas [INFO]: ciclo nro. 25
facu 01/05/2016 15:48:30 RecibirOfertas [INFO]: ciclo nro. 26
facu 01/05/2016 15:48:40 RecibirOfertas [INFO]: ciclo nro. 27
facu 01/05/2016 15:48:50 RecibirOfertas [INFO]: ciclo nro. 28
facu 01/05/2016 15:49:00 RecibirOfertas [INFO]: ciclo nro. 29
facu 01/05/2016 15:49:10 RecibirOfertas [INFO]: ciclo nro. 30
```

Archivo: 3303_20160420.csv

```
$ head -n 10 3303_20160420.csv
```

```
     7900028    12000

     7914112    5000

     7938136    45000

     7962072    40130

     7962042    41000

     7977120    15000
```

```
     7985071    10000

     7993127    15000

     7993037    20000

     7995069    66000
```

```
Archivo: 3402_20160420.csv
$ head -n 10 3402_20160420.csv
     7897075    35000

     7900069    4000

     7913140    40000

     7914127    35000

     7920088    50000

     7920024    55000

     7922053    30000

     7929018    25000

     7929016    25001

     7933111    132500
```

- Imprima el log de ProcesarOfertas y los primeros 10 registros de cada uno de los archivos de resultado.

```
$ cat LOGDIR/ProcesarOfertas.log
facu 01/05/2016 15:44:19 ProcesarOfertas [INFO]: Inicio de ProcesarOfertas
facu 01/05/2016 15:44:19 ProcesarOfertas [INFO]: Cantidad de archivos a procesar:  2
facu 01/05/2016 15:44:19 ProcesarOfertas [INFO]: Archivo a procesar: 3303_20160420.csv
facu 01/05/2016 15:44:20 ProcesarOfertas [INFO]: El archivo
"/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/aceptados/3303_20160420.csv" se movió
exitosamente al destino
"/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/procesados/procesadas".
facu 01/05/2016 15:44:20 ProcesarOfertas [INFO]: Registros leidos = 93: cantidad de
ofertas validas = 49 cantidad de ofertas rechazadas = 44
facu 01/05/2016 15:44:20 ProcesarOfertas [INFO]: Archivo a procesar: 3402_20160420.csv
facu 01/05/2016 15:44:24 ProcesarOfertas [INFO]: El archivo
"/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/aceptados/3402_20160420.csv" se movió
exitosamente al destino
"/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/procesados/procesadas".
```

```
facu 01/05/2016 15:44:24 ProcesarOfertas [INFO]: Registros leidos = 312: cantidad de
ofertas validas 174 cantidad de ofertas rechazadas = 138
facu 01/05/2016 15:44:24 ProcesarOfertas [INFO]: cantidad de archivos procesados 2
facu 01/05/2016 15:44:24 ProcesarOfertas [INFO]: cantidad de archivos rechazados 0
facu 01/05/2016 15:44:24 ProcesarOfertas [INFO]: Fin de ProcesarOfertas

$ head -n 10 PROCDIR/validas/20160519.txt
3303;20160420;7914112;7914;112;5000;OVEJERO,GENARO;facu;01/05/2016 15:44:19
3303;20160420;7938136;7938;136;45000;LIBERONA,CESAR;facu;01/05/2016 15:44:19
3303;20160420;7962042;7962;042;41000;BRAVO,BRENDA;facu;01/05/2016 15:44:19
3303;20160420;7977120;7977;120;15000;GRAZIANI,IVANA;facu;01/05/2016 15:44:19
3303;20160420;7985071;7985;071;10000;MORENO,EDITA;facu;01/05/2016 15:44:19
3303;20160420;7993037;7993;037;20000;RAMIREZ,ANA;facu;01/05/2016 15:44:19
3303;20160420;7995069;7995;069;66000;ROMERO,MARIA;facu;01/05/2016 15:44:19
3303;20160420;8069058;8069;058;35006;DANI,NOEMI;facu;01/05/2016 15:44:19
3303;20160420;8107051;8107;051;39025;CHOQUE,ZAMBRANO;facu;01/05/2016 15:44:19
3303;20160420;8139077;8139;077;20000;MERCADO,BERNARDO;facu;01/05/2016 15:44:19

$ head -n 10 PROCDIR/procesadas/3303_20160420.csv
7900028;12000
7914112;5000
7938136;45000
7962072;40130
7962042;41000
7977120;15000
7985071;10000
7993127;15000
7993037;20000
7995069;66000

$ head -n 10 PROCDIR/procesadas/3402_20160420.csv
7897075;35000
7900069;4000
7913140;40000
7914127;35000
7920088;50000
7920024;55000
7922053;30000
7929018;25000
7929016;25001
7933111;132500

$ head -n 10 PROCDIR/rechazadas/3303.rech
3303_20160420.csv;No alcanza el monto Minimo;7900028,12000;facu;01/05/2016 15:44:19
3303_20160420.csv;Suscriptor no puede participar;7962072,40130;facu;01/05/2016 15:44:19
3303_20160420.csv;Suscriptor no puede participar;7993127,15000;facu;01/05/2016 15:44:19
3303_20160420.csv;Supera el monto maximo;8041127,197500;facu;01/05/2016 15:44:19
```

```
3303_20160420.csv;Supera el monto maximo;8065039,50030;facu;01/05/2016 15:44:19
3303_20160420.csv;Supera el monto maximo;8083002,50020;facu;01/05/2016 15:44:19
3303_20160420.csv;Supera el monto maximo;8127166,70500;facu;01/05/2016 15:44:19
3303_20160420.csv;Supera el monto maximo;8162112,95000;facu;01/05/2016 15:44:19
3303_20160420.csv;Supera el monto maximo;8191071,40010;facu;01/05/2016 15:44:19
3303_20160420.csv;Supera el monto maximo;8209076,105000;facu;01/05/2016 15:44:19

$ head -n 10 PROCDIR/rechazadas/3402.rech
3402_20160420.csv;No alcanza el monto Minimo;7900069,4000;facu;01/05/2016 15:44:20
3402_20160420.csv;Supera el monto maximo;7914127,35000;facu;01/05/2016 15:44:20
3402_20160420.csv;Supera el monto maximo;7933111,132500;facu;01/05/2016 15:44:20
3402_20160420.csv;Suscriptor no puede participar;7968056,30200;facu;01/05/2016 15:44:20
3402_20160420.csv;Suscriptor no puede participar;7972079,63500;facu;01/05/2016 15:44:20
3402_20160420.csv;Supera el monto maximo;8001109,90000;facu;01/05/2016 15:44:20
3402_20160420.csv;Supera el monto maximo;8001112,90000;facu;01/05/2016 15:44:20
3402_20160420.csv;Supera el monto maximo;8001099,101000;facu;01/05/2016 15:44:20
3402_20160420.csv;Supera el monto maximo;8005120,160500;facu;01/05/2016 15:44:20
3402_20160420.csv;Supera el monto maximo;8022056,65000;facu;01/05/2016 15:44:20

NOKDIR/ -> No contiene archivos.
```

- Ejecute GenerarSorteo e imprima el archivo de salida

```
$ bash LanzarProceso.sh "GenerarSorteo.sh"
$ cat PROCDIR/sorteos/1_20160519.srt
1;92
2;134
3;163
4;109
5;43
6;161
7;6
8;165
9;1
10;16
11;19
12;66
13;47
14;42
15;80
16;167
17;60
18;125
19;75
20;133
21;13
```

22;77
23;151
24;118
25;17
26;130
27;113
28;61
29;120
30;62
31;138
32;96
33;95
34;142
35;164
36;39
37;99
38;115
39;44
40;38
41;9
42;106
43;105
44;55
45;83
46;147
47;128
48;24
49;32
50;22
51;139
52;64
53;49
54;137
55;158
56;153
57;159
58;126
59;114
60;122
61;58
62;154
63;117
64;33
65;119
66;107
67;15

68;104
69;20
70;73
71;112
72;26
73;168
74;21
75;156
76;40
77;157
78;18
79;56
80;148
81;103
82;90
83;7
84;160
85;71
86;152
87;141
88;123
89;93
90;11
91;53
92;97
93;28
94;86
95;27
96;50
97;67
98;23
99;124
100;76
101;111
102;69
103;34
104;129
105;68
106;81
107;74
108;10
109;4
110;91
111;52
112;35
113;45

114;98
115;143
116;94
117;140
118;136
119;59
120;54
121;72
122;127
123;46
124;51
125;5
126;29
127;110
128;155
129;70
130;78
131;150
132;116
133;121
134;135
135;31
136;84
137;12
138;88
139;87
140;146
141;166
142;82
143;57
144;162
145;65
146;102
147;149
148;144
149;85
150;89
151;101
152;8
153;63
154;25
155;30
156;2
157;108
158;41
159;37

```
160;100
161;36
162;145
163;132
164;3
165;79
166;131
167;48
168;14
```

- 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.

```
$ ./DeterminarGanadores.pl -a
Esta es la ayuda del comando DeterminarGanador del módulo CIPAK.
Previo a ejecutarse este comando, debe inicializarse correspondientemente el ambiente
Para saber cómo iniciar correctamente el ambiente, diríjase a la documentación
proporcionada
El módulo puede ejecutarse con los siguientes argumentos:
 './DeterminarGanadores.pl -g' graba los resultados en el directorio
/home/facu/Escritorio/grupo05/CIPAK_G5/Grupo05/informes/
'./DeterminarGanadores.pl' sólo se muestran las consultas en pantalla
Una vez lanzado el módulo, se presentará la pantalla de bienvenida
La misma le pedirá que ingrese una opción para acceder a una consulta
Se tienen las siguientes consultas disponibles:
A entrega el numero ganador del sorteo correspondiente a la fecha de adjudicación
indicada por parámetro
B indica, dentro de los grupos especificados por el usuario, qué numero de orden
resulto ganador del sorteo
C indica, dentro de los grupos especificados por el usuario, qué numero de orden
resulto ganador de la licitación, tomando en cuenta al ganador del sorteo
D indica, para un grupo en particular, los ganadores por sorteo y licitación
El ingreso de cualquier otro caracter diferente a los mencionados, causará el término
de ejecución del módulo

Posibles Errores del módulo (cualquiera de estos errores provoca el paro en la
ejecucion del comando):
1. No se inicializó el ambiente. Primero inicialicelo
Causa del error: No se ejecutó el comando PrepararAmbiente, por lo cual el módulo no
puede ser ejecutado
2. Ya se está corriendo una instancia de DeterminarGanadores
```

Causa del error: Se intenta ejecutar, otra vez y en otra ventana, el módulo cuando el mismo ya está siendo ejecutado

```
$ ./DeterminarGanadores.pl
Bienvenido al CIPAK.
Para consultar por un sorteo, ingrese A
Para consultar por los ganadores de un sorteo dentro de uno o varios grupos, ingrese B
Para consultar los ganadores por licitacion dentro de uno o varios grupos, ingrese C
Para consultar los ganadores por licitacion y sorteo dentro de un grupo, ingrese D
Para salir, presione enter
Seleccione el tipo de consulta que quiera realizar
```

```
$ ./DeterminarGanadores.pl -g
Bienvenido al CIPAK.
Para consultar por un sorteo, ingrese A
Para consultar por los ganadores de un sorteo dentro de uno o varios grupos, ingrese B
Para consultar los ganadores por licitacion dentro de uno o varios grupos, ingrese C
Para consultar los ganadores por licitacion y sorteo dentro de un grupo, ingrese D
Para salir, presione enter
Script ejecutado con opción de grabado de archivos
Seleccione el tipo de consulta que quiera realizar
```

```
$ ./DeterminarGanadores.pl
Bienvenido al CIPAK.
Para consultar por un sorteo, ingrese A
Para consultar por los ganadores de un sorteo dentro de uno o varios grupos, ingrese B
Para consultar los ganadores por licitacion dentro de uno o varios grupos, ingrese C
Para consultar los ganadores por licitacion y sorteo dentro de un grupo, ingrese D
Para salir, presione enter
Seleccione el tipo de consulta que quiera realizar
A
Se le presentara un listado de adjudicaciones válidas.

2_20160519.srt
1_20160519.srt
Especifique el ID del sorteo que quiere mostrar por pantalla
1
Especifique la fecha, en formato AAAAMMDD
20160519
Nro de Sorteo 001 le correspondió al número de orden 144
Bienvenido al CIPAK.
Para consultar por un sorteo, ingrese A
Para consultar por los ganadores de un sorteo dentro de uno o varios grupos, ingrese B
Para consultar los ganadores por licitacion dentro de uno o varios grupos, ingrese C
Para consultar los ganadores por licitacion y sorteo dentro de un grupo, ingrese D
Para salir, presione enter
```

```
Seleccione el tipo de consulta que quiera realizar
B
2_20160519.srt
1_20160519.srt
Especifique el ID del sorteo que quiere mostrar por pantalla
2
Especifique la fecha, en formato AAAAMMDD
20160519
Ingrese el o los grupos que quiera consultar. De querer un rango, ingrese xxxx-yyyy
ATENCION: Si usted está consultando un grupo en particular, debe ingresar uno solo
7900-7910
Ganadores del sorteo 2 de fecha 20160519
Ganador por sorteo del grupo 7900: Número de orden 006 ,CRIVELLI,WALTER(Numero de
sorteo 002)
Ganador por sorteo del grupo 7901: Número de orden 086 ,DOMINGUEZ,MARCELO(Numero de
sorteo 002)
Ganador por sorteo del grupo 7902: Número de orden 052 ,ESPINDOLA,NADIA(Numero de
sorteo 002)
Ganador por sorteo del grupo 7906: Número de orden 006 ,GALLARDO,JOSE(Numero de sorteo
002)
Ganador por sorteo del grupo 7907: Número de orden 081 ,GRIFFA,DIEGO(Numero de sorteo
002)
Bienvenido al CIPAK.
Para consultar por un sorteo, ingrese A
Para consultar por los ganadores de un sorteo dentro de uno o varios grupos, ingrese B
Para consultar los ganadores por licitacion dentro de uno o varios grupos, ingrese C
Para consultar los ganadores por licitacion y sorteo dentro de un grupo, ingrese D
Para salir, presione enter
Seleccione el tipo de consulta que quiera realizar
C
Ganadores por Licitacion 2 de fecha 20160519 .
2_20160519.srt
1_20160519.srt
Especifique el ID del sorteo que quiere mostrar por pantalla
1
Especifique la fecha, en formato AAAAMMDD
20160519
Ingrese el o los grupos que quiera consultar. De querer un rango, ingrese xxxx-yyyy
ATENCION: Si usted está consultando un grupo en particular, debe ingresar uno solo
7900-7910
Ganador por licitación del grupo 7900 : Número de orden 006.CRIVELLI,WALTER con $ 25000
(Nro de Sorteo 001)
Ganador por licitación del grupo 7901 : Número de orden 094.DOTTI,CRISTIAN con $ 50550
(Nro de Sorteo 001)
Ganador por licitación del grupo 7902 : Número de orden 052.ESPINDOLA,NADIA con $ 47501
(Nro de Sorteo 001)
```

```
Ganador por licitación del grupo 7906 : Número de orden 006.GALLARDO,JOSE con $ 41496
(Nro de Sorteo 001)
Ganador por licitación del grupo 7907 : Número de orden 088.GRIVA,JUAN con $ 21463,42
(Nro de Sorteo 001)
Bienvenido al CIPAK.
Para consultar por un sorteo, ingrese A
Para consultar por los ganadores de un sorteo dentro de uno o varios grupos, ingrese B
Para consultar los ganadores por licitacion dentro de uno o varios grupos, ingrese C
Para consultar los ganadores por licitacion y sorteo dentro de un grupo, ingrese D
Para salir, presione enter
Seleccione el tipo de consulta que quiera realizar
D
2_20160519.srt
1_20160519.srt
Especifique el ID del sorteo que quiere mostrar por pantalla
1
Especifique la fecha, en formato AAAAMMDD
20160519
Ingrese el o los grupos que quiera consultar. De querer un rango, ingrese xxxx-yyyy
ATENCION: Si usted está consultando un grupo en particular, debe ingresar uno solo
7907
7907 - 081 S (GRIFFA,DIEGO)
7907 - 088 L (GRIVA,JUAN)
```

APÉNDICE A

CARÁTULA

Tema	Grupo	Ayudante	Correcciones			
			Fecha	Hora Inicio	Hora Fin	Resultado
K	1 3 5 7 9		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, maximo?				
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 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.

Licitar 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 Grupoxx 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
- ARRIDIR
- **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 vacio, 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 adjudicacion **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

Licitar 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 adjudicacion **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
- Sorteo_Id 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 adjudicacion **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.

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

Titulo: 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

Titulo: 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")

Titulo: 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, vacio;
- 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 numero 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 dónde 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, auditoria 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 preveer 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, señalizar 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: dia/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