LABORATORIO 3 de PROGRAMACIÓN 2
Curso 2010

AeroProg2

Sistema de administración de aerolíneas
Segunda Parte

Índice

1 Introducción....2
2 Implementación....2
2.1 Comandos que cambian....3
2.2 Nuevos comandos....6
2.3 Persistencia....9
3 Formato de una condición....13
4 Módulos de definición....14
5 Librerías Comando e ImpresionResultado....14
6 Dependencia de Módulos....15
7 Estimativo de Dificultad....15
8 Nueva información y comunicación....16
8.1 Uso del foro....16
9 Entrega....17
9.1 Entrega Laboratorio 3....17
9.2 Plazos de entrega....18
9.3 Forma de entrega....18
9.4 Identificación de los archivos de las entregas....18
9.5 Re-entregas....18
9.6 Clave de acceso para entregar....18
9.7 Verificación de lo entregado....19
10 Individualidad....19
11 Compilador....19
12 Recursos del lenguaje....19
13 Evaluación....19
14 Anexo....20

1 Introducción

Este laboratorio consiste en incorporar nuevas funcionalidades al laboratorio anterior. Se agregan funcionalidades de lectura y escritura de archivos para registrar el estado de la aerolínea en un momento dado. Se modifican listados existentes y se agregan nuevos.

2 Implementación

En las siguientes sub-secciones se describen los nuevos comandos a implementar y aquellos que modifican su comportamiento.

Si en un comando falla más de un control, se imprime el mensaje de error del primer control fallado que aparece en la letra para ese comando. Es decir, que si por ejemplo fallan los controles a) y c) de un comando se debe imprimir el mensaje de error a).

Los ejemplos que se incluyen para cada comando son incrementales a partir de la siguiente inicialización de la aerolínea:

```
Bienvenidos a AeroProg2
version 2.0
InCo-FIng-UdelaR

>altaCliente 'Roberto' 'Gomez' 1863456 UY
Cliente 1863456-UY ingresado.
>altaCliente 'Patricia' 'della Giovampaola' 4533567 IT
Cliente 4533567-IT ingresado.
>altaAvion 10 100 AIRBUSA300
Avion 0 ingresado.
>altaAvion 20 250 BOEING747
Avion 1 ingresado.
>altaLinea INCO100 0 MVD EZE 1400 45 DIARIA
Linea INCO100 ingresada.
>altaLinea INCO101 0 EZE MVD 1600 45 DIARIA
Linea INCO101 ingresada.
>altaLinea INCO237 1 MVD MAD 2030 720 SEMANAL VIERNES
Linea INCO237 ingresada.
>iniciar 20100424
Fecha 20100424 inicializada.
>reserva INCO101 20100426 1863456 UY TURISTA 3
Asiento 3-TURISTA-20100426-INCO101 resevado a 1863456-UY.
>reserva INCO101 20100427 1863456 UY TURISTA
Asiento 1-TURISTA-20100427-INCO101 resevado a 1863456-UY.
>reserva INCO237 20100813 4533567 IT PRIMERA 12
Asiento 12-PRIMERA-20100813-INCO237 resevado a 4533567-IT.
>compra INCO100 20100426 1863456 UY EFECTIVO TURISTA
Asiento 1-TURISTA-20100426-INCO100 vendido a 1863456-UY.
Costo: 60-EFECTIVO. Puntos: 7.
>compra INCO101 20100427 1863456 UY EFECTIVO TURISTA 1
Asiento 1-TURISTA-20100427-INCO101 vendido a 1863456-UY.
Costo: 60-EFECTIVO. Puntos: 7.
>cancela INCO237 20100813 4533567 IT PRIMERA 12
Reserva 12-PRIMERA-20100813-INCO237-4533567-IT cancelada.
>siguienteDia
Fecha actual 20100425. Reservas liberadas: 1.
```

2.1 Comandos que cambian

Información de Línea – infoLinea *idLinea*

Descripción: Se imprime en pantalla información de la línea pasada como parámetro.

Asumir

- Los parámetros tienen los formatos descritos en el laboratorio anterior.

Control

a) Existe una línea en el sistema con identificador *idLinea*.

Mensaje de error

a) ERROR: No existe una linea con identificador *idLinea*.

Acción

- Se imprime en pantalla:

  Linea: *idLinea-idAvion-aeroOrigen-aeroDestino-horaSalida-duracion-frecuencia*.

Cambio

- No se imprimen todos los datos del avión, sólo su identificador.

Ejemplo

```
>infoLinea INCO100
Linea: INCO100-0-MVD-EZE-1400-45-DIARIA.
>infoLinea INCO237
Linea: INCO237-1-MVD-MAD-2030-720-SEMANAL VIERNES.
```

Información de Vuelo – infoVuelo *idLinea fecha*

Descripción: Se imprime en pantalla información de compras y reservas del vuelo pasado como parámetro.

Asumir

- Los parámetros tienen los formatos descritos en el laboratorio anterior.

Control

a) El comando iniciar se había ejecutado previamente.

b) Existe una línea en el sistema con identificador *idLinea*.

c) La línea *idLinea* tiene un vuelo que sale en la fecha *fecha*.

d) La fecha es posterior al día actual.

Mensaje de error

a) ERROR: Fecha no inicializada.

b) ERROR: No existe una linea con identificador *idLinea*.

c) ERROR: No existe vuelo *fecha -idLinea*.

d) ERROR: Comando invalido para la fecha *fecha*.

Acción

- Se imprime en pantalla:

Vuelo: *idLinea-fecha.*

Primera:

p1...p10

...

pn-10...pn

Turista:

t1...t10

...

tm-10...tm

- *pi* y *ti* son, para las clases primera y turista respectivamente, el estado actual del asiento *i* siguiendo la siguiente codificación:
  - Libre: L
  - Reservado: `R-pasaporte`*`Cliente-nacionalidadCliente-`*
  - Comprado: `C-`*`pasaporteCliente-nacionalidadCliente-`*

- El estado de los asientos se imprime de hasta 10 por línea. La cantidad total de asientos no tiene por qué ser múltiplo de 10, en ese caso la última línea tendrá menos de 10 asientos.

Cambio

- Se imprime al principio "Vuelo: *idLinea-fecha.*".
- En caso de reservado o comprado se imprime la información del cliente.

Ejemplo

```
>infoVuelo INCO101 20100426
Vuelo: INCO101-20100426
Primera:
LLLLLLLLLL
Turista:
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
>infoVuelo INCO101 20100427
Vuelo: INCO101-20100427
Primera:
LLLLLLLLLL
Turista:
C-1863456-UY-LLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
```

Listar Líneas – listarLineas ordenarPor ordenForma

Descripción: Se listan en pantalla los datos de las líneas de la aerolínea en el orden especificado en *ordenarPor,* donde para cada campo el orden será ascendente o descendente según se especifica en el parámetro *ordenForma* a través de los valores **ASC** o **DESC** respectivamente.

Asumir:

- Los valores que puede tomar *ordenarPor* son:
  - **ID**: representa que se quiere ordenar por el identificador de la línea.
  - **AVION**: representa que se quiere ordenar por el identificador del avión.
  - **ORIGEN**: representa que se quiere ordenar por el aeropuerto origen.
  - **DESTINO**: representa que se quiere ordenar por el aeropuerto destino.
  - **HORA**: representa que se quiere ordenar por la hora de salida.
  - **DURACION:** representa que se quiere ordenar por la duración.
- Se incluye al menos el nombre de un campo en *ordenarPor*.
- Los valores que toma *ordenForma* son ***ASC*** y/o ***DESC***.
- La cantidad de valores ***ASC*** o ***DESC*** especificados en el parámetro *ordenForma*, coincide con la cantidad de campos especificados en el parámetro *ordenarPor*
- Los nombres de los campos en *ordenarPor* y los valores en *ordenForma* se separan por el carácter ':' (dos puntos) y se corresponden uno a uno, es decir, el nombre del i-ésimo campo en *ordenarPor* se corresponde con el i-ésimo valor de *ordenForma*.

Acción

- Se imprime en pantalla:

  Lineas:

  Linea: *idLinea1-idAvion1-descripcionAvion1-aeroOrigen1-aeroDestino1-horaSalida1-duracion1-frecuencia1*.

  ...

  Linea: *idLineaN-idAvionN-descripcionAvionN-aeroOrigenN-aeroDestinoN-horaSalidaN-duracionN-frecuenciaN*.

  - donde las líneas *idLinea1* ... *idLineaN* tienen el orden especificado por *ordenarPor* y *ordenForma*.

Ejemplo

```
>altaAvion 5 50 ATR42
Avion 2 ingresado.
>altaLinea INCO102 1 MVD EZE 1400 45 DIARIA
Linea INCO102 ingresada.
>altaLinea IncO103 2 MVD EZE 1400 60 DIARIA
Linea IncO103 ingresada.
>listarLineas ID ASC
Lineas:
Linea: INCO100-0-MVD-EZE-1400-45-DIARIA.
Linea: INCO101-0-EZE-MVD-1600-45-DIARIA.
Linea: INCO102-1-MVD-EZE-1400-45-DIARIA.
Linea: INCO237-1-MVD-MAD-2030-720-SEMANAL VIERNES.
Linea: IncO103-2-MVD-EZE-1400-60-DIARIA.
>listarLineas HORA DESC
Lineas:
Linea: INCO237-1-MVD-MAD-2030-720-SEMANAL VIERNES.
Linea: INCO101-0-EZE-MVD-1600-45-DIARIA.
Linea: INCO100-0-MVD-EZE-1400-45-DIARIA.
```

```
Linea: INCO102-1-MVD-EZE-1400-45-DIARIA.
Linea: IncO103-2-MVD-EZE-1400-60-DIARIA.
>listarLineas HORA:ID DESC:ASC
Lineas:
Linea: INCO237-1-MVD-MAD-2030-720-SEMANAL VIERNES.
Linea: IncO103-2-MVD-EZE-1400-60-DIARIA.
Linea: INCO100-0-MVD-EZE-1400-45-DIARIA.
Linea: INCO101-0-EZE-MVD-1600-45-DIARIA.
Linea: INCO102-1-MVD-EZE-1400-45-DIARIA.
>listarLineas HORA:DURACION:AVION:ORIGEN:DESTINO ID:DESC:ASC:DESC:ASC:ASC:ASC
Lineas:
Linea: INCO237-1-MVD-MAD-2030-720-SEMANAL VIERNES.
Linea: IncO103-2-MVD-EZE-1400-60-DIARIA.
Linea: INCO100-0-MVD-EZE-1400-45-DIARIA.
Linea: INCO101-0-EZE-MVD-1600-45-DIARIA.
Linea: INCO102-1-MVD-EZE-1400-45-DIARIA.
```

2.2 Nuevos comandos

Comentario - # *'texto'*

Descripción: Permite ingresar un comentario.

Asumir

- *texto* contiene solamente caracteres alfanuméricos, espacios en blanco y signos de puntuación.
- El largo máximo que puede tomar *'texto'* se encuentro definido en la constante *MAXLARGOPARAM* en el módulo Comando.

Acción

- Se imprime el texto ingresado.

Ejemplo

```
>#  'comentario'
'comentario'
```

Información de Avión – infoAvion *idAvion*

Descripción: Se imprime en pantalla información del avión pasado como parámetro.

Asumir

- Los parámetros tienen los formatos descritos en el laboratorio anterior,

Control

a) Existe un avión en el sistema con identificador *idAvion*.

Mensaje de error

a) ERROR: No existe un avion con identificador *idAvion*.

Acción

- Se imprime en pantalla:

  Avion: *idAvion-cantAsientosPrimera -cantAsientosTurista-descripcionAvion*.

Ejemplo

```
>infoAvion 0
Avion: 0-10-100-AIRBUSA300.
>infoAvion 2
Avion: 2-5-50-ATR42.
```

Filtrar Líneas – filtrarLineas condicionFiltro

Descripción: Se listan en pantalla los datos de las líneas de la aerolínea que cumplen con la *condicionFiltro* ordenadas en el orden en que fueron ingresadas al sistema.

El formato de la condición se presenta en la sección 3 .

Asumir

- La condición *condicionFiltro* es correcta.

Acción

- Se imprime en pantalla:

  Lineas:

  Linea: *idLinea1-idAvion1-descripcionAvion1-aeroOrigen1-aeroDestino1-horaSalida1-duracion1-frecuencia1*.

  ...

  Linea: *idLineaN-idAvionN-descripcionAvionN-aeroOrigenN-aeroDestinoN-horaSalidaN-duracionN-frecuenciaN*.

  - donde las líneas *idLinea1 ... idLineaN* cumplen con *condicionFiltro* y se encuentran en el orden en que fueron insertadas al sistema.

Ejemplo

```
>filtrarLineas ( ( ID = INCO100 ) OR ( NOT ( HORA > 1800 ) ) )
Lineas:
Linea: INCO100-0-MVD-EZE-1400-45-DIARIA.
Linea: INCO101-0-EZE-MVD-1600-45-DIARIA.
Linea: INCO102-1-MVD-EZE-1400-45-DIARIA.
Linea: IncO103-2-MVD-EZE-1400-60-DIARIA.
>filtrarLineas ()
Lineas:
Linea: INCO100-0-MVD-EZE-1400-45-DIARIA.
Linea: INCO101-0-EZE-MVD-1600-45-DIARIA.
Linea: INCO237-1-MVD-MAD-2030-720-SEMANAL VIERNES.
Linea: INCO102-1-MVD-EZE-1400-45-DIARIA.
Linea: IncO103-2-MVD-EZE-1400-60-DIARIA.
>filtrarLineas ( () AND ( ORIGEN < MVD ) )
Lineas:
Linea: INCO101-0-EZE-MVD-1600-45-DIARIA.
>filtrarLineas  ( ( ORIGEN = CUN ) OR ( DIA = VIERNES ) )
Lineas:
Linea: INCO237-1-MVD-MAD-2030-720-SEMANAL VIERNES.
```

Buscar Combinaciones – buscarCombinaciones *origen destino fecha clase*

Descripción: Se imprime en pantalla la información de cómo llegar de un aeropuerto *origen* a un aeropuerto *destino*, partiendo el día de la *fecha* pasada como parámetro, ordenando la información en orden creciente por el precio total de los vuelos combinados para la clase indicada en el parámetro *clase*.

La forma de llegar de un aeropuerto al otro no necesariamente se pueda realizar utilizando una línea, sino una combinación de varias de ellas. Se listarán los tramos, identificación de línea, fecha y hora de partida de las combinaciones posibles.

En caso que dos combinaciones coincidan en el precio, se desempata en orden lexicográfico creciente de los identificadores de la primera línea de cada combinación. En caso de coincidir en la primera línea se desempata por la segunda y así sucesivamente hasta lograr una diferencia.

Las posibles combinaciones tienen las siguientes restricciones:

1) No se puede repetir una línea. Es decir que no pueden haber combinaciones "circulares" del tipo: INCO100 - INCO101 - INCO100.
2) El máximo tiempo de espera permitido entre la llegada de un vuelo y la salida del siguiente es de 12 horas. Es decir que el cliente no debe tener que esperar por un trasbordo más de 12 horas.
3) Tiene que haber al menos un asiento disponible en cada vuelo de la combinación en la clase *clase*.

Asumir

- Los parámetros tienen los formatos descritos en el laboratorio anterior

Control

a) La fecha *fecha* tiene que ser al menos un día posterior a la fecha actual. Es decir que si por ejemplo, la fecha actual es 20100424 sólo se puede buscar combinaciones de vuelos que salgan a partir del 20100425 en adelante.

Mensaje de error

a) ERROR: Comando invalido para la fecha *fecha*.

Acción

- Se imprime en pantalla:

  Posibles combinaciones:

  Combinacion *1*: *precio1*

  idlinea11-aeroOrigen11-aeroDestino11-fechaSalida11-horaSalida11-fechaLlegada11-horaLlegada11

  ...

  idlinea1J -aeroOrigen1J-aeroDestino1J-fechaSalida1J-horaSalida1J-fechaLlegada1J-horaLlegada1J

  ...

  Combinacion *N*: *precioN*

  idlineaN1-aeroOrigenN1-aeroDestinoN1-fechaSalidaN1-horaSalidaN1-fechaLlegadaN1-horaLlegadaN1

  ...

  idlineaNK-aeroOrigenNK-aeroDestinoNK-fechaSalidaNK-horaSalidaNK-fechaLlegadaNK-horaLlegadaNK

Ejemplo

```
>altaAvion 200 480 AIRBUS380
Avion 3 ingresado.
>altaAvion 1 1 BoeingX32
Avion 4 ingresado.
```

```
>altaLinea INCO104 3 MVD CUN 1000 300 DIARIA
Linea INCO104 ingresada.
>altaLinea INCO105 3 CUN EZE 1600 360 DIARIA
Linea INCO105 ingresada.
>altaLinea INCO106 3 CUN SCL 1800 300 SEMANAL MARTES
Linea INCO106 ingresada.
>altaLinea INCO107 4 SCL CUN 1800 120 DIARIA
Linea INCO107 ingresada.
>altaLinea INCO108 4 SCL EZE 2330 30 DIARIA
Linea INCO108 ingresada.
>altaLinea INCO109 3 MVD SCL 2330 30 SEMANAL MIERCOLES
Linea INCO109 ingresada.
>reserva INCO108 20100427 1863456 UY TURISTA
Asiento 1-TURISTA-20100427-INCO108 resevado a 1863456-UY.
>#  'ejecuto buscarCombinaciones'
'ejecuto buscarCombinaciones'
>buscarCombinaciones MVD EZE 20100427 TURISTA
Combinacion 1: 3600
INCO100-MVD-20100427-14:00-EZE-20100427-14:45
Combinacion 2: 3600
INCO102-MVD-20100427-14:00-EZE-20100427-14:45
Combinacion 3: 4800
IncO103-MVD-20100427-14:00-EZE-20100427-15:00
Combinacion 4: 52800
INCO104-MVD-20100427-10:00-CUN-20100427-15:00
INCO105-CUN-20100427-16:00-EZE-20100427-22:00
```

2.3 Persistencia

El formato de los archivos de persistencia es el siguiente:

datosAviones

datosLineas

datosClientes

fechaHoy

vuelos

donde:

datosAviones tiene formato:

A

Avion: *cantAsientosPrimera1-cantAsientosTurista1-descripcionAvion1*.

…

Avion: *cantAsientosPrimeraA-cantAsientosTuristaA-descripcionAvionA*.

Con *A* la cantidad de aviones de la aerolínea.

datosLineas tiene formato:

L

Linea: *idLinea1-idAvion1*-aeroOrigen*1-aeroDestino1-horaSalida1-duracion1-frecuencia1*.

```
…
Linea: idLineaL-idAvionL-aeroOrigenL-aeroDestinoL-horaSalidaL-duracionL-frecuenciaL.
```

Con *L* la cantidad de líneas de la aerolínea. Las líneas se listan en el orden de ingreso al sistema.

datosClientes tiene formato:

```
C
Cliente:'nombre1'-'apellido1'-pasaporte1-nacionalidad1-puntos1.
Movimientos:
asiento11-clase11-fecha11-idLinea11-estado11
…
asiento1N-clase1N-fecha1N-idLinea1N-estado1N
finCliente
…
Cliente:'nombreC'-'apellidoC'-pasaporteC-nacionalidadC-puntosC.
Movimientos:
asientoC1-claseC1-fechaC1-idLineaC1-estadoC1
…
asientoCM-claseCM-fechaCM-idLineaCM-estadoCM
finCliente
```

Con *C* la cantidad de clientes de la aerolínea y, por ejemplo, *N* y *M* la cantidad de movimientos realizados por los clientes *1* y *C* respectivamente.

fechaHoy tiene formato *aaaammdd*

vuelos tiene formato:

```
V
Vuelo: idLinea1-fecha1.
Primera:
p11...p110
…
p1(N-10) … p1N
Turista:
t11 … t110
…
t1(M-10)...t1M
finVuelo
…
```

```
Vuelo: idLineaV-fechaV.
Primera:
p11 … p110
…
p1(J-10) … p1J
Turista:
t11 … t110
…
t1(K-10) … t1K
finVuelo
```

Con *V* la cantidad de vuelos que salen a partir de *fechaHoy* (inclusive) que tienen al menos un asiento reservado o comprado. *N* y *M* la cantidad de asientos de primera y turista respectivamente del avión de la línea del vuelo *1*. *J* y *K* la cantidad de asientos de primera y turista respectivamente del avión de la línea del vuelo *V*.

Orden creciente por fecha, en caso de empate por hora y en caso de empate por el orden de ingreso al sistema.

Cada *pij* y *tij* puede ser:

- L
- R-pasaporte*Cliente-nacionalidadCliente-*
- C-*pasaporteCliente-nacionalidadCliente-*

Persistir – persistir *nombreArchivo*

Descripción: Persiste en almacenamiento secundario el estado de la aerolínea en el archivo de nombre *nombreArchivo*. La ruta del archivo (incluyendo su nombre) viene dada en el parámetro *nombreArchivo*. El formato del archivo es el descripto anteriormente.

Asumir

- El archivo de nombre *nombreArchivo* referencia a una ruta válida y sin espacios en blanco .

Control

a) El comando iniciar se había ejecutado previamente.

Mensaje de error

a) ERROR: Fecha no inicializada.

Acción

- Se guarda en el archivo *nombreArchivo* la aerolínea con el formato especificado.
- Si el archivo no existe se crea, y si existe se sobrescribe.
- Imprime en pantalla:

  Aerolinea persistida en fecha *fecha*.

  -*fecha* es la fecha actual.

Ejemplo

```
>persistir c:\pluna.aer
Aerolinea persistida en fecha 20100425.
```

El archivo persistido es el siguiente:

```
5
Avion: 0-10-100-AIRBUSA300.
Avion: 1-20-250-BOEING747.
Avion: 2-5-50-ATR42.
Avion: 3-200-480-AIRBUS380.
Avion: 4-1-1-BoeingX32.
11
Linea: INCO100-0-MVD-EZE-1400-45-DIARIA.
Linea: INCO101-0-EZE-MVD-1600-45-DIARIA.
Linea: INCO237-1-MVD-MAD-2030-720-SEMANAL VIERNES.
Linea: INCO102-1-MVD-EZE-1400-45-DIARIA.
Linea: IncO103-2-MVD-EZE-1400-60-DIARIA.
Linea: INCO104-3-MVD-CUN-1000-300-DIARIA.
Linea: INCO105-3-CUN-EZE-1600-360-DIARIA.
Linea: INCO106-3-CUN-SCL-1800-300-SEMANAL MARTES.
Linea: INCO107-4-SCL-CUN-1800-120-DIARIA.
Linea: INCO108-4-SCL-EZE-2330-30-DIARIA.
Linea: INCO109-3-MVD-SCL-2330-30-SEMANAL MIERCOLES.
2
Cliente: 'Roberto'-'Gomez'-1863456-UY-7.
Movimientos:
3-TURISTA-20100426-INCO101-MULTA
1-TURISTA-20100427-INCO101-COMPRA
1-TURISTA-20100426-INCO100-COMPRA
1-TURISTA-20100427-INCO108-RESERVA
finCliente
Cliente: 'Patricia'-'della Giovampaola'-4533567-IT-0.
Movimientos:
12-PRIMERA-20100813-INCO237-CANCELA
finCliente
20100425
3
Vuelo: INCO100-20100426.
Primera:
LLLLLLLLL
Turista:
C-1863456-UY-LLLLLLLLL
LLLLLLLLL
LLLLLLLLL
LLLLLLLLL
LLLLLLLLL
LLLLLLLLL
LLLLLLLLL
LLLLLLLLL
LLLLLLLLL
LLLLLLLLL
finVuelo
Vuelo: INCO101-20100427.
Primera:
LLLLLLLLL
Turista:
C-1863456-UY-LLLLLLLLL
LLLLLLLLL
LLLLLLLLL
LLLLLLLLL
LLLLLLLLL
LLLLLLLLL
LLLLLLLLL
LLLLLLLLL
LLLLLLLLL
LLLLLLLLL
finVuelo
Vuelo: INCO108-20100427.
Primera:
L
Turista:
R-1863456-UY-
finVuelo
```

Recuperar – recuperar *nombreArchivo*

Descripción: Substituye el estado actual de la aerolínea por uno creado a partir de un archivo que sigue el formato preestablecido.

Asumir

- El archivo a leer viene en el formato correcto.
- La ruta y el nombre del archivo son correctos.

Acción

- Si existe una aerolínea cargada al momento de invocar el comando se debe destruir la misma.
- Crea la aerolínea a partir del contenido del archivo.
- La fecha del sistema es la indicada en el archivo.
- Imprime en pantalla:

  Aerolinea recuperada con fecha *fecha*.

  - *fecha* es la fecha que el archivo cargado tiene como *fechaHoy*.

Ejemplo

```
>recuperar c:\pluna.aer
Aerolinea recuperada con fecha 20100425.
```

3 Formato de una condición

Las condiciones son expresiones que se evalúan para cada linea de la aerolínea, y devuelven verdadero si satisfacen la condición y falso en caso contrario. Es decir, que las condiciones son usadas para filtrar líneas. El comando que involucra condiciones es listarLineas.

Las condiciones pueden ser de la siguiente forma:

- **()**: esta condición implica que la condición es *true* para toda línea.
- **(nombre operador valor)** donde:
  - *nombre* representa el atributo de la línea sobre el que se quiere realizar la comparación con *valor*. Los posibles valores para *nombre* se describen al final de esta sección.
  - *operador* puede ser:
    - >
    - <
    - =

    Se utilizará el orden lexicográfico habitual para la comparación entre strings.
    En caso de comparar la frecuencua o el día de la semana solo es permitido el operador =.
  - *valor* puede ser:
    - Un valor numérico
    - Una cadena de caracteres (sin espacios)
    - Un literal con la frecuencia (DIARIA, SEMANAL)
    - Un literal con el día de la semana (LUNES MARTES MIERCOLES JUEVES VIERNES SABADO DOMINGO)
- **(condicion operador_binario condicion)** donde:
  - *condicion* es cualquier condición válida.
  - *operador_binario* puede ser:

        - **AND**
        - **OR**
- **(operador_unario condicion)** donde:
    - *condicion* es cualquier condición válida.
    - *operador_unario* puede ser:
        - **NOT**

Entre *nombre* y *operador* solo hay un espacio en blanco de separación, al igual que entre *operador* y *valor*.
Luego del paréntesis '(' hay un espacio en blanco de separación y antes del paréntesis ')' también hay un espacio en blanco de separación.

Nombres válidos para una condición del tipo (nombre operador valor):

- **ID**: representa que se quiere evaluar la condición por el identificador de la línea.
- **AVION**: representa que se quiere evaluar la condición por el identificador del avión.
- **ORIGEN**: representa que se quiere evaluar la condición por el aeropuerto origen.
- **DESTINO**: representa que se quiere evaluar la condición por el aeropuerto destino.
- **HORA**: representa que se quiere evaluar la condición por la hora de salida.
- **DURACION**: representa que se quiere evaluar la condición por la duración.
- **FRECUENCIA**: representa que se quiere evaluar la condición por la frecuencia.
- **DIA**: representa que se quiere evaluar la condición por el día de la semana.

Ejemplos de condiciones:

- `( ( ID = 'INCO100' ) OR ( NOT ( HORA > 1800 ) ) )`
- `()`
- `( () AND ( ORIGEN < 'MVD' ) )`

Si la frecuencia es diaria, cualquier día debe evaluar a verdadero, por lo tanto la condición:

- `( DIA = LUNES )`

equivale a:

- `( ( FRECUENCIA = DIARIA ) OR ( ( FRECUENCIA = SEMANAL ) AND ( DIA = LUNES ) ) )`

4 Módulos de definición

El estudiante debe obtener los archivos correspondientes a los módulos de definición (**.def*) en un archivo *.zip* desde la sección Laboratorio del sitio web del curso: ***http://www.fing.edu.uy/inco/cursos/prog2***

Los módulos de definición no se pueden modificar. Tenga en cuenta que para el proceso de corrección se utilizarán los archivos *.def* publicados.

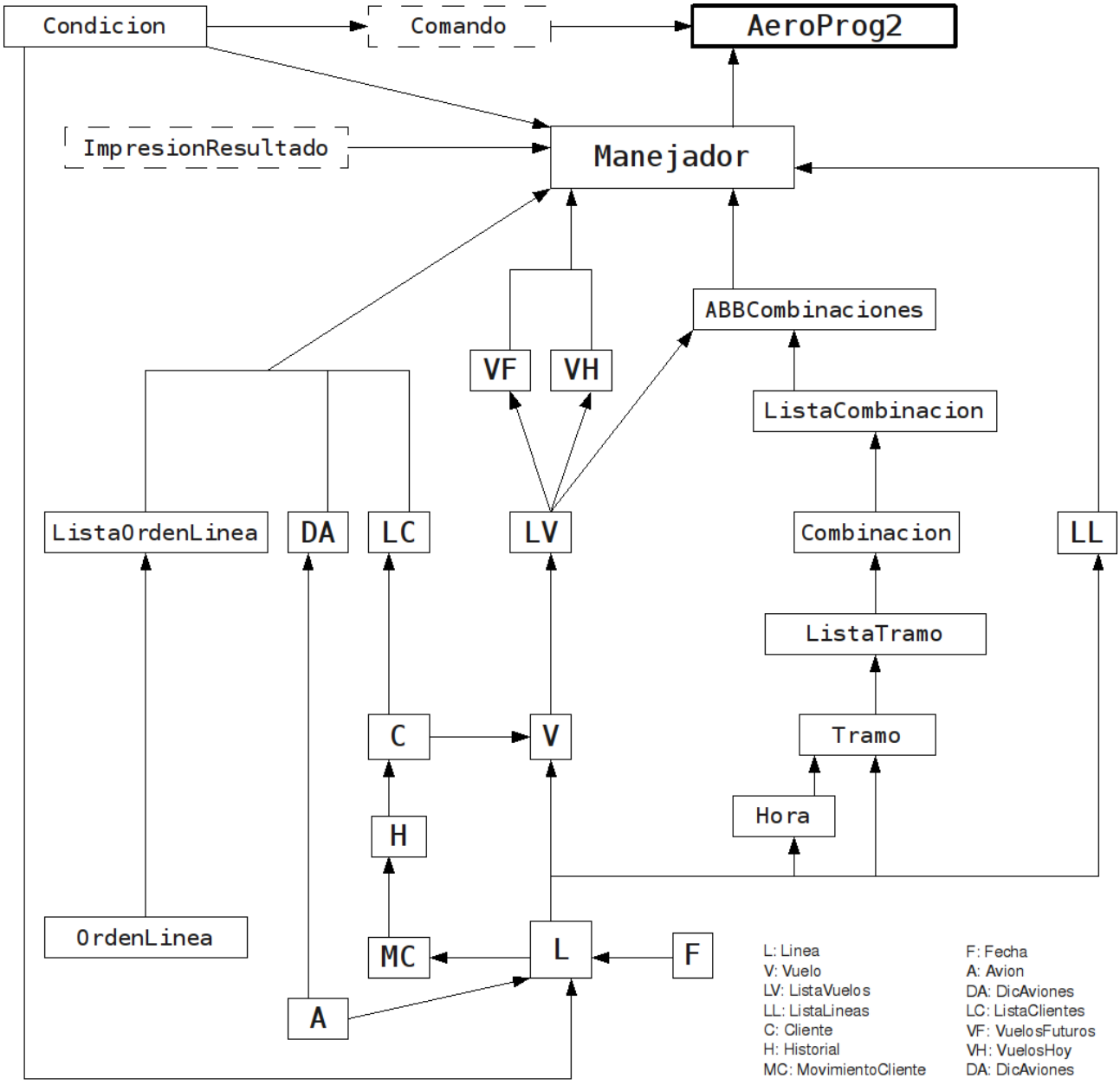
Las funciones, procedimientos y tipos de datos de los módulos de implementación deben coincidir con las definiciones correspondientes que se encuentran en los módulos de definición tal como aparecen descriptos en estos.

5 Librerías Comando e ImpresionResultado

El estudiante debe obtener las librerías (**.lib*) para la impresión de mensajes éxito y errores (ver su descripción en *ImpresionResultado.def*) y el módulo Comando (ver su descripción en *Comando.def*) que procesa la entrada de los comandos y sus parámetros, en un archivo *zip* desde la sección Laboratorio del sitio web del curso.

Estos módulos se entregan ya compilados, es decir, prontos para ser linkeditados con los restantes módulos del laboratorio.

6 Dependencia de Módulos



7 Estimativo de Dificultad

En la siguiente tabla se indica para cada módulo a implementar un nivel de dificultad. Esta información puede resultar de utilidad al estudiante para planificar su trabajo en el laboratorio.

Se asignó el valor 1 a los módulos de menor dificultad y 10 a los de mayor dificultad. Luego se ubicaron en el intervalo los restantes módulos teniendo en cuenta su nivel de dificultad.

Nivel de Dificultad Laboratorio 2	Módulos
1	Avion, DicAviones, Fecha, Historial, ListaClientes, ListaLineas, ListaVuelos, MovimientoCliente, VuelosFuturos, VuelosHoy
2	AeroProg2, Cliente, Hora, Linea, OrdenLinea, Tramo, Vuelo
3	ListaCombinacion, ListaOrdenLinea, ListaTramo
4	ABBCombinacione, Condicion
5	Combinacion
10	Manejador

Observación: muchos de los módulos que se encuentran en un mismo nivel tienen un alto grado de similitud.

8 Nueva información y comunicación

En caso de ser necesario se publicarán documentos con los agregados y/o correcciones al laboratorio que puedan surgir con el avance del curso.

Toda publicación de nueva información se realizará en la sección laboratorio del sitio *web* del curso, y será debidamente avisada en las clases teóricas, prácticas y de laboratorio, foro y página *web* del curso.

Los estudiantes cuentan con las clases de consulta de laboratorio y el foro, para discutir acerca del laboratorio.

Las casillas personales de los docentes no son el medio de comunicación adecuado para este tipo de discusiones, por lo cual se solicita no utilizarlo.

8.1 Uso del foro

Antes de usar el foro del curso, es obligación leer el Reglamento de participación en los foros.
Cada mensaje debe ser enviado al foro que corresponda (dudas de letra, casos de test, programa principal, los distintos módulos, etc).

Para consultar dudas relacionadas a los casos de pruebas que se publiquen, el formato a utilizar para definir el asunto de un nuevo mensaje, es el siguiente:

- [nroCasodePrueba] Descripción breve

Antes de publicar un mensaje nuevo **verificar si ya no hay un hilo de conversación para el módulo, operación y descripción que se quiera consultar**.
Si lo hay entonces continuar sobre el hilo existente.
Sino iniciar un nuevo hilo según el formato definido anteriormente.

Al responder un mensaje, **NO INCLUIR** el texto que contiene el mensaje al que se responde, se puede leer el anterior siguiendo la conversación. Con esto se evita la acumulación y el tráfico de datos innecesarios, facilitando también la lectura de los mensajes.

En caso de no respetar lo indicado anteriormente los docentes no contestarán dichos mensajes.

9 Entrega

9.1 Entrega Laboratorio 3

En esta entrega el estudiante deberá realizar la implementación de los siguientes módulos.

Módulos del laboratorio anterior:

- Avion.mod
- Fecha.mod
- DicAviones.mod
- Linea.mod
- MovimientoCliente.mod
- Historial.mod
- Cliente.mod
- Vuelo.mod
- ListaLineas.mod
- ListaClientes.mod
- ListaVuelos.mod
- VuelosFuturos.mod
- VuelosHoy.mod
- Manejador.mod

Módulos nuevos:

- ABBCombinaciones.mod
- Combinacion.mod
- Condicion.mod
- Hora.mod
- ListaCombinacion.mod
- ListaOrdenLinea.mod
- ListaTramo.mod
- OrdenLinea.mod
- Tramo.mod
- Valor.mod

y el programa principal *AeroProg2.mod*.

El estudiante debe obtener los archivos correspondientes a los módulos de definición (*.def), para esta entrega en un archivo *zip*, desde la sección laboratorio del sitio *web* del curso.

Para cada módulo se deberá implementar el tipo de datos necesario para la representación así como las operaciones definidas en el mismo.

El comportamiento de entrada y salida debe ser exactamente igual al descrito en la letra del laboratorio.
No debe cambiarse la sintaxis de los comandos ni de las respuestas a los mismos. Téngase en cuenta que esto será considerado en la evaluación, como un factor crítico para la aprobación del laboratorio.

No se pueden realizar módulos adicionales a los solicitados.

También se podrá entregar un archivo de texto con los comentarios que el estudiante crea necesarios. Este archivo deberá llamarse **`leame.txt`**.

El estudiante que no respete alguna de las indicaciones anteriores asume el riesgo de que su trabajo no sea (o no pueda ser) corregido y que en consecuencia se invalide la entrega, con la consiguiente pérdida del curso.

9.2 Plazos de entrega

El plazo para la entrega es hasta el 21 de Junio del 2010, hasta las 8:00 horas.

Los trabajos deberán ser entregados dentro de los plazos indicados anteriormente. No se aceptarán entregas de trabajos fuera de fecha y hora. La no entrega o la entrega fuera de los plazos indicados, implica la pérdida del curso.

9.3 Forma de entrega

Las entregas se realizarán de forma automática vía *web* como ya se hizo en el laboratorio 2. Para ello, se deberá acceder a la sección laboratorio del sitio *web* del curso y seguir los pasos que se explicarán en la página correspondiente.

Es importante señalar que para realizar las entregas se solicitará al estudiante la clave que utiliza para realizar los trámites de Bedelía.

Los estudiantes que se inscribieron en **forma individual** deberán ingresar la clave de bedelía para efectuar la entrega.
Los estudiantes que se inscribieron en **forma grupal**, para efectuar la entrega, deberán ingresar la clave de bedelía del integrante del grupo que se inscribió como representante, el otro estudiante no estará habilitado para realizar entregas.

Se recuerda que las computatodars de las salas 114 y 115 del 1er piso, así como las de las bandejas metálicas del Cuerpo Norte de Facultad tienen conexión a la red. Por lo cual, se puede acceder desde ellos a la página *web* para realizar las entregas.

El estudiante que no respete alguna de las indicaciones que se detallan a continuación asume el riesgo que su trabajo no sea (o no pueda ser) corregido y que en consecuencia se invalide la entrega, con la consiguiente pérdida del curso.

9.4 Identificación de los archivos de las entregas

Cada uno de los archivos a entregar debe contener en la primera línea del archivo un comentario con el número de cédula del estudiante **sin el guión y sin dígito de verificación**. Por ejemplo, si un estudiante tiene número de cédula 1910876-4 deberá escribir en la primera línea de cada uno de los archivos entregados el siguiente comentario:

```
(* 1910876 *)
IMPLEMENTATION MODULE nombreAr;
```

En el ejemplo anterior se asumió que se trata del archivo *nombreAr.mod*.
En caso de tratarse de grupos de dos integrantes se deben incluir ambas cédulas separadas por un espacio en blanco.

9.5 Re-entregas

Desde el día que se habilite el receptor de entrega del laboratorio y hasta la fecha de culminación de la entrega, el estudiante podrá realizar todas las re-entregas que considere necesario, teniendo en cuenta que sólo la última entrega será corregida.
Se recomienda realizar una primera entrega 'de prueba' de forma de verificar que puede realizar el procedimiento con normalidad. Recuerde que solo la última entrega será corregida.

9.6 Clave de acceso para entregar

Verifique que tiene o dispone de la clave que le entregó Bedelía antes de realizar la 1ª entrega. En caso de no tenerla

comuníquese con Bedelía.

9.7 Verificación de lo entregado

La responsabilidad de verificar la integridad de cada entrega es del estudiante.

10 Individualidad

No existirán instancias en el curso para reclamos frente a la detección por parte de los docentes de trabajos de laboratorio no individuales, independientemente de las causas que pudiesen originar la no individualidad (a modo de ejemplo y sin ser exhaustivos: utilización de código realizado en cursos anteriores u otros cursos, perder el código, olvidarse del código en lugares accesibles a otros estudiantes, prestar el código o dejar que el mismo sea copiado por otros estudiantes, dejar la terminal con el usuario abierto al retirarse, enviarse código por mail, etc.), es decir, que se considera a cada estudiante RESPONSABLE DE SU TRABAJO DE LABORATORIO Y DE QUE EL MISMO SEA INDIVIDUAL.
Los trabajos de laboratorio que a juicio de los docentes no sean individuales **serán eliminados, con la consiguiente pérdida del curso**, para **todos** los involucrados. Además todos los casos serán enviados a los órganos competentes de la Facultad, lo cual puede acarrear sanciones de otro carácter y gravedad para los estudiantes involucrados.
Asimismo **se prohíbe el envío de código vinculado al laboratorio al foro del curso**, dado que el mismo será considerado como una forma de compartir código y será sancionado de forma severa.

11 Compilador

Todos los módulos entregados deben compilar y linkeditar con el compilador del curso (XDS-Modula2). De no ser así el trabajo será eliminado, con la consiguiente pérdida del curso.

El estándar a utilizar es el estándar ISO.

12 Recursos del lenguaje

Se podrán utilizar todas las herramientas del lenguaje que sean vistas en el curso, tanto en los teóricos como en los prácticos.

13 Evaluación

Se evaluará positivamente:
- Modularidad y estructuración del código.
- Cantidad y calidad de los comentarios.
- Claridad en el código.
- Uso de estructuras de datos adecuadas al problema.
- Nombre de las variables y constantes acordes a la función que desempeñan en el programa.

Se penalizará fuertemente:
- Uso de programación no estructurada.
- Uso de variables globales dentro de funciones o procedimientos.
- Programas que funcionen incorrectamente.

La tarea se evaluará con los ejemplos presentados en la letra del laboratorio, así como también con otros casos de prueba. En todos los casos el programa deberá funcionar correctamente de acuerdo a la especificación proporcionada.

Por último se recomienda que el estudiante tenga presente que: l**a entrega de módulos y/o programas que, como mínimo, no compilen o linkediten, o cuya salida, en cuanto a su resultado, no sea exactamente idéntica a la de los ejemplos presentados en la letra del laboratorio, serán eliminados, con la consiguiente pérdida del curso.** Se recuerda que las tareas deben **compilar con el**

compilador del curso (*Native XDS-x86* para Windows) y **funcionar en las máquinas de los laboratorios de facultad** (Windows XP).

14 Anexo

A continuación se muestra la correspondencia de los mensajes de éxito y de error según el código en el TAD *ImpresionResultado* agrupados por comando para los comandos nuevos. Las palabras del mensaje que se encuentran en itálica, son parámetros variables, es decir, su valor depende sobre qué o quién se este aplicando el comando. En color celeste se marcan los mensajes de éxito para cada comando.

Comando	**Mensaje**	**TAD ImpresionResultado: *CodigoImpresionResultado***
infoAvion	ERROR: No existe un avion con identificador *idAvion*.	`ERROR_AVION_NO_EXISTE`
buscarCombinaciones	ERROR: Comando invalido para la fecha *fecha*.	`ERROR_FECHA_TARDIA`
persistir	ERROR: Fecha no inicializada.	`ERROR_FECHA_NO_INICIALIZADA`
	Aerolinea persistida en fecha *fecha*.	`PERSISTIR`
recuperar	Aerolinea recuperada con fecha *fecha*.	`RECUPERAR`