LABORATORIO 2 de PROGRAMACIÓN 2
Curso 2010

AeroProg2

Sistema de administración de aerolíneas
Primera Parte

Índice

1 Introducción

Se desea implementar un sistema para la administración de una aerolínea. Esta aerolínea divide su administración en tres grandes grupos de funcionalidades. El primer grupo de funcionalidades tiene como objetivo permitir a la aerolínea la gestión de su negocio, entendiéndose como esto el ingreso de clientes, aviones y líneas al sistema.

En el contexto de este laboratorio, un cliente es aquella persona que usa los servicios de la aerolínea; un avión es un determinado vehículo de transporte aéreo con características particulares como cantidad de asientos de la categoría "primera" y "turista"; y una línea representa un tramo entre dos destinos que recorre un avión. Además cada línea tiene una frecuencia, hora de salida y duración.

Para que el sistema pueda operar correctamente necesita tener configurada la fecha del día, pues es en función de ella que se realizan los controles de cumplimiento de las reglas de la aerolínea que se describen mas adelante. Para este propósito el sistema cuenta con la funcionalidad de inicializar la fecha del sistema.

El segundo grupo de funcionalidades permite la operación de la aerolínea, es decir la reserva, compra y cancelación de pasajes en los vuelos ofrecidos por la aerolínea. Un vuelo es un tramo entre dos destinos que recorre un avión, con determinada duración y hora de salida en una fecha determinada. En otras palabras es un línea que volará en una fecha específica.
Las funcionalidades de reserva, compra y cancelación de pasajes en los vuelos operan de acuerdo a las reglas de la aerolínea que se describen a continuación:

REGLAS DE LA AEROLÍNEA

RESERVA:
1) Se pueden reservar pasajes de un vuelo hasta dos días antes del día de su salida.
2) Sólo se pueden reservar asientos que no se encuentren comprados o reservados.
3) Al comenzar el día anterior a la salida de un vuelo se penaliza a todos los clientes que tienen reservas en el mismo con la quita de 10 o 15 puntos por cada hora de duración del vuelo en turista o primera, respectivamente. Todas las reservas quedan liberadas para habilitar su compra directa.

COMPRA:
4) Se pueden comprar pasajes de un vuelo hasta el día anterior a su salida.
5) Sólo se pueden comprar asientos que no se encuentren reservados por otro cliente o comprados.
6) La compra se puede realizar en efectivo o con puntos (si el cliente dispone de los puntos suficientes). El precio del pasaje es:
- PRIMERA: 120 dólares o 150 puntos por cada hora de duración del vuelo.
- TURISTA: 80 dólares o 100 puntos por cada hora de duración del vuelo.

7) Al comprar un pasaje el cliente gana 10 o 15 puntos por cada hora de duración del vuelo en turista o primera, respectivamente.
8) La numeración de los asientos va desde 1 a la cantidad de asientos del avión para cada clase de asiento (turista, primera)

Los cálculos de puntos y efectivo se redondean siempre a valores enteros (hacia arriba).

CANCELACION:
9) Se pueden cancelar pasajes de un vuelo hasta dos días antes del día de su salida.

Estas operaciones solo se podrán ejecutar si el sistema tiene configurada la fecha del día. Por otra parte y para su correcto funcionamiento, es necesario que el sistema pueda "avanzar" la fecha que tiene configurada. Para esto el sistema tiene la funcionalidad de avanzar la fecha del día a su siguiente día.

Por ultimo, el tercer grupo de funcionalidades esta formado por las funcionalidades que brindan información sobre la

gestión y operación de la aerolínea. Por ejemplo, incluye operaciones que permiten desplegar la información de un vuelo dado por una línea y una fecha particular, o desplegar información de un cliente particular, entre otras.

Las operaciones de los tres grupos de funcionalidades mencionadas anteriormente se presentan y especifican a continuación agrupadas respectivamente en las sub-secciones: *Comandos de Gestión*, *Comandos de Operación* y *Comandos de Información.*.

El objetivo de este laboratorio es implementar el sistema ***AeroProg2***, nombre que la aerolínea le ha dado al sistema para la administración de su aerolínea.

2 Implementación

El programa principal del sistema tendrá como nombre ***AeroProg2***. Al inicio de la ejecución de ***AeroProg2*** debe aparecer en la pantalla el siguiente mensaje:

```
Bienvenidos a AeroProg2
version 1.0
InCo-FIng-UdelaR

>
```

3 Intérprete de Comandos

El programa ***AeroProg2*** lee comandos desde la entrada estandar y los ejecuta. La sintaxis general de los comandos es la siguiente:

```
nombreComando parámetros ...
```

Un comando está formado por un nombre seguido de una lista de parámetros eventualmente vacía. El nombre del comando está separado de los parámetros por **un (1)** espacio. Asimismo, cada parámetro se separa del siguiente por **un (1)** espacio.

Téngase en cuenta que:

1. Se puede asumir que **TODOS** los comandos se ingresan con una sintaxis correcta (nombre de comando, cantidad de parámetros, largo de los parámetros, tipo de datos, etc).

2. Se puede asumir que **TODOS** los identificadores que se pasan como parámetro a los distintos comandos son sintácticamente correctos de acuerdo a la definición dada para los mismos.

3. En las siguientes sub-secciones, para cada comando se detalla lo siguiente:

   - **Descripción:** descripción general del comando.
   - **Asumir**: lo que se puede asumir (no es necesario chequearlo) para un comando.
   - **Control**: lo que la operación debe controlar para que tenga efecto en el sistema, dejándola en un estado consistente. Si algún control no se cumple, la operación no tendrá efecto, desplegando un mensaje de error.
   - **Mensajes de Error**: mensajes que deben mostrarse en caso de fallar alguno de los controles.
   - **Acción:** el efecto de la operación sobre el sistema.
   - **Ejemplo:** un ejemplo exitoso del uso del comando. Los ejemplos que se incluyen para cada comando en las siguientes secciones, son incrementales, es decir, el ejemplo de un comando tiene como estado inicial del sistema el estado del ejemplo del comando anterior que se encuentra en la letra del laboratorio.

   Nota: los mensajes que se imprimen en la salida estándar no llevan tildes, ya sea en los casos de éxito como de error.

4. A cada **Control** le corresponde un único mensaje de error. El mensaje de error a mostrar para un control, se

indica en *Mensajes de Error* en el mismo orden de aparición que en *Control*. Por ejemplo, el mensaje de error del segundo control del comando, se corresponde al segundo mensaje de error en el ítem *Mensajes de Error*.

En las siguientes sub-secciones se describen los comandos a implementar.

3.1 Operaciones sobre el sistema

3.1.1 Comandos de Gestión

Alta de Cliente - altaCliente *'nombre' 'apellido' pasaporte nacionalidad*

Descripción: Agrega un cliente al sistema con los siguientes datos: nombre, apellido, pasaporte y nacionalidad. El cliente se identifica a través de su pasaporte y nacionalidad.

El uso de nombres y apellidos compuestos está permitido, y se encuentra delimitado entre comillas simples. Ejemplo de nombre compuesto *'nombre'* : 'Juan Pablo'.

Asumir

- Los parámetros *nombre* y *apellido* están compuestos únicamente por letras de la A-Z, a-z y el espacio en blanco.
- El espacio en blanco se utiliza como separador en caso de que el nombre o apellido sea compuesto.
- El largo máximo que pueden tomar *nombre*, *apellido*, *pasaporte* y *nacionalidad* se encuentra definido en las constantes correspondientes del módulo Cliente.
- Los valores para los campos *nombre* y *apellido* vienen delimitados por comillas simples.
- El parámetro *pasaporte* puede estar compuesto por números y/o letras.
- El parámetro *nacionalidad* es una cadena de dos letras que codifica cada nacionalidad de acuerdo al estándar ISO 3166-1 alfa-2 (http://es.wikipedia.org/wiki/ISO_3166-1).

Control

a) No existe un cliente en el sistema con el mismo pasaporte y nacionalidad.

Mensaje de error

a) ERROR: Existe un cliente con pasaporte *pasaporte* y nacionalidad *nacionalidad*.

Acción

- Se agrega el cliente al sistema de la aerolínea con los datos pasados como parámetros.
- El puntaje del cliente se inicia en 0 y su historial de movimientos vacío.
- Se imprime en pantalla:
  Cliente *pasaporte – nacionalidad* ingresado.

Ejemplo

```
> altaCliente 'Roberto' 'Gomez' 1863456 UY
Cliente 1863456-UY ingresado.
> altaCliente 'Patricia' 'della Giovampaola' 4533567 IT
```

Cliente 4533567-IT ingresado.

Alta de Avión - altaAvion *cantAsientosPrimera cantAsientosTurista descripcionAvion*

Descripción: Agrega un avión al sistema con la cantidad de asientos en primera, la cantidad de asientos en clase turista y la descripción.

Asumir

- El valor máximo que pueden tomar *cantAsientosPrimera* y *cantAsientosTurista* se encuentra definido en las constantes correspondientes del módulo Avion.
- El parámetro *descripcionAvion* está compuesto por letras y/o números.
- El largo máximo que puede tomar *descripcionAvion* se encuentra definido en la constante correspondiente en el módulo Avion.
- La cantidad de aviones en el sistema es menor que la constante MAX_CANTIDAD_AVIONES, definida en el módulo DicAviones.

Acción

- Se agrega el avión al sistema de la aerolínea con la capacidad pasada como parámetro.
- Se le asigna un identificador entero secuencial comenzando en 0. Por lo tanto el identificador de un avión que se ingresa es igual a la cantidad de aviones que existian previamente en el sistema.
- Se imprime en pantalla:
  Avion *n* ingresado.

  - *n* es el identificador asignado.

Ejemplo

> altaAvion 10 100 `AIRBUSA300`

Avion 0 ingresado.
> altaAvion 20 250 `BOEING747`
Avion 1 ingresado.

Alta de Línea - altaLinea *idLinea idAvion aeroOrigen aeroDestino horaSalida duracion frecuencia*

Descripción: Agrega una línea al sistema con los siguientes datos: identificador de la línea, aeropuertos de origen y destino, hora de salida, duración del vuelo y frecuencia.

Asumir

- El largo máximo que pueden tomar *idLinea, aeroOrigen* y *aeroDestino* se encuentra definido en las constantes correspondientes del módulo Linea.
- El parámetro *idLinea* se compone por letras y/o números.
- Los parámetros *aeroOrigen* y *aeroDestino* se componen de tres letras, codificando cada aeropuerto de

acuerdo al código IATA (http://www.world-airport-codes.com/).

- Los parámetros *idAvion* y *duracion* son valores enteros.
- El parámetro *duración* representa los minutos de duración del vuelo.
- El valor de *horaSalida* es una cadena con formato *hhmm* que representa una hora válida (ej: 2030 representa 20:30 hs).
- Si el parámetro *idavion* es el identificador de un avión insertado al sistema, el avión estará siempre disponible en el aeropuerto de origen a la hora de salida.
- El parámetro *frecuencia* puede ser DIARIA o SEMANAL. Si la frecuencia es SEMANAL se agrega un octavo parámetro *dia*, que puede ser: LUNES, MARTES, MIERCOLES, JUEVES, VIERNES, SABADO o DOMINGO.

Control

a) No existe una línea en el sistema con el mismo identificador.

b) El avión asociado a la línea ha sido insertado al sistema.

Mensaje de error

a) ERROR: Existe una linea con identificador *idLinea*.

b) ERROR: No existe un avion con identificador *idavion*.

Acción

- Se agrega la línea al sistema con los valores pasados como parámetro.
- Se imprime en pantalla:
  Linea *idLinea* ingresada.

Ejemplo

```
> 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.
```

Inicializar Fecha de Sistema - iniciar *fecha*

Descripción: Inicializa la fecha actual en el sistema de la aerolínea. Esto habilita la posterior ejecución de los comandos de Operación.

Asumir

- El valor de *fecha* es una cadena con formato *aaaammdd* que representa una fecha válida (ej: 20100426 representa 26 de mayo de 2010).

Control

a) El comando iniciar no se había ejecutado previamente.

Mensaje de error

a) ERROR: Fecha ya inicializada.

Acción

- Se inicializa la fecha actual del sistema como *fecha*. Se habilita la ejecución de los comandos de Operación.
- Se imprime en pantalla:
  Fecha *fecha* inicializada.

Ejemplo

```
> iniciar 20100424
Fecha 20100424 inicializada.
```

3.1.2 Comandos de Operación

Reserva de Pasaje – reserva *idLinea fecha pasaporteCliente nacionalidadCliente clase [asiento]*

Descripción: El sistema reserva para el cliente de pasaporte *pasaporteCliente* y nacionalidad *nacionalidadCliente* un asiento de la clase *clase* en el vuelo dado por la línea *idLinea* y fecha *fecha.* El parámetro *asiento* es opcional. En caso de incluirlo como parámetro, el sistema intenta reservar ese número de asiento, en caso contrario intenta reservar el primer número de asiento disponible para la clase especificada.

Asumir

- Los parámetros tienen los formatos descritos en los comandos anteriores.
- El parámetro *clase* puede ser PRIMERA o TURISTA.

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) El cliente con pasaporte *pasaporteCliente* y nacionalidad *nacionalidadCliente* existe en el sistema.
e) Si se ingresa el parámetro *asiento*, el mismo debe ser mayor que 0 y menor o igual a la cantidad de asientos para la clase *clase* en el avión asociado a la línea *idLinea*.
f) Si se ingresa el parámetro *asiento*, el mismo no debe estar actualmente comprado ni reservado (regla 2 de la aerolínea).
g) En caso de no pasar el parámetro *asiento,* la cantidad de asientos disponibles (ni comprados ni reservados) de la clase *clase* en el vuelo dado por la línea *idLinea* y fecha *fecha* tiene que ser mayor que 0.
h) La fecha *fecha* tiene que ser al menos dos dias posterior a la fecha actual (regla 1 de la aerolínea). Es decir que si, por ejemplo, la fecha actual es 20100424 sólo se puede reservar asientos en vuelos que salgan a partir del 20100426 en adelante.

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: Cliente *pasaporteCliente-nacionalidadCliente* no ingresado.

e) ERROR: Asiento *asiento-clase-idLinea* invalido.

f) ERROR: Asiento *asiento-clase-fecha-idLinea* no disponible.

g) ERROR: No hay asientos disponibles en *clase-fecha-idLinea.*

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

Acción

- Si se incluye el asiento y está disponible, se realiza la reserva del mismo.
- Si NO se incluye el asiento y hay alguno disponible se le da el de número más chico posible, y se le asigna la reserva.
- En caso de poder realizar la reserva: a) se cambia el estado del asiento a reservado en el vuelo correspondiente b) se genera un nuevo movimiento con estado RESERVA para el cliente dado, especificando los datos de la reserva. El mismo formará parte de su historial de movimientos.
- Se imprime en pantalla:
  Asiento *num-clase-fecha-idLinea* reservado a *pasaporteCliente-nacionalidadCliente.*

  - *num* es el número de asiento reservado.

Ejemplo

```
> reserva INCO101 20100426 1863546 UY TURISTA 3
Asiento 3-TURISTA-20100427-INCO101 resevado a 1863546-UY.
> reserva INCO101 20100427 1863546 UY TURISTA
Asiento 1-TURISTA-20100427-INCO101 resevado a 1863546-UY.
> reserva INCO237 20100813 4533567 IT PRIMERA 12
Asiento 12-PRIMERA-20100813-INCO237 resevado a 4533567-IT.
```

Compra de Pasaje – compra *idLinea fecha pasaporteCliente nacionalidadCliente modoPago clase [asiento]*

Descripción: El sistema vende al cliente de pasaporte *pasaporteCliente* y nacionalidad *nacionalidadCliente* un pasaje en la clase *clase* en el vuelo dado por la línea *idLinea* y fecha *fecha.* El parámetro *asiento* es opcional. En caso de incluirlo como parámetro, el sistema intenta vender ese número de asiento, en caso contrario intenta vender el primer número de asiento disponible para la clase especificada.

Asumir

- Los parámetros tienen los formatos descritos en los comandos anteriores.
- El parámetro *modoPago* puede ser EFECTIVO o PUNTOS.

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) El cliente con pasaporte *pasaporteCliente* y nacionalidad *nacionalidadCliente* existe en el sistema.
e) Si se ingresa el parámetro *asiento*, el mismo debe ser mayor que 0 y menor o igual a la cantidad de asientos para la clase *clase* en el avión asociado a la línea *idLinea*.
f) Si se ingresa el parámetro *asiento*, el mismo no debe estar actualmente reservado por otro cliente ni comprado (regla 5 de la aerolínea).
g) En caso de no pasar el parámetro *asiento,* la cantidad de asientos disponibles (ni comprados ni reservados) de la clase *clase* en el vuelo dado por la línea *idLinea* y fecha *fecha* tiene que ser mayor que 0.
h) La fecha *fecha* tiene que ser al menos un día posterior a la fecha actual (regla 4 de la aerolínea). Es decir que si, por ejempo, la fecha actual es 20100424 sólo se puede comprar asientos en vuelos que salgan a partir del 20100425 en adelante.
i) Si el *modoPago* es PUNTOS el cliente con pasaporte *pasaporteCliente* y nacionalidad *nacionalidadCliente* debe tener la cantidad de PUNTOS suficientes (sin contar con los puntos ganados por ese vuelo) para cubrir el pago total del pasaje.

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: Cliente *pasaporteCliente-nacionalidadCliente* no ingresado.
e) ERROR: Asiento *asiento-clase-idLinea* inválido.
f) ERROR: Asiento *asiento-clase-fecha-idLinea* no disponible.
g) ERROR: No hay asientos disponibles en *clase-fecha-idLinea.*
h) ERROR: Comando invalido para la fecha *fecha*.
i) ERROR: Saldo de puntos del cliente *pasaporteCliente-nacionalidadCliente* es insuficiente.

Acción

- Si se incluye el parámetro *asiento* y está disponible, o reservado por el cliente con pasaporte *pasaporteCliente* y nacionalidad *nacionalidadCliente*, se realiza la venta del mismo.
- Si NO se incluye el parámetro *asiento* y hay alguno disponible se le vende el de número más chico posible.
- En caso de que el asiento vendido haya sido reservado previamente por el cliente, se modifica su historial cambiando el estado del movimiento original de RESERVA a COMPRA.
- En caso de que el asiento vendido NO haya sido reservado previamente por el cliente, se modifica su historial creando un nuevo movimiento con información del vuelo y estado COMPRA.
- Se calcula el precio del pasaje de acuerdo al *modoPago* y a la regla 6 de la aerolínea. En caso de pagar con PUNTOS, los mismos se restan a los puntos del cliente.
- Se calculan los puntos obtenidos por la compra del pasaje, según la regla 7 de la aerolínea, y se los suma a los puntos del cliente.

- Se imprime en pantalla:
  Asiento *num-clase-fecha-idLinea* vendido a *pasaporteCliente-nacionalidadCliente.*
  Costo: *costo-modoPago.* Puntos: *puntos.*

  - *num* es el número de asiento vendido.
  -*costo* es el costo calculado del pasaje.
  -*puntos* son los puntos ganados por la compra del pasaje.

Ejemplo

> compra INCO100 20100426 1863546 UY EFECTIVO TURISTA
Asiento 1-TURISTA-20100426-INCO100 vendido a 1863546-UY.
Costo: 60-EFECTIVO. Puntos: 7.
> compra INCO101 20100427 1863546 UY EFECTIVO TURISTA 1
Asiento 1-TURISTA-20100427-INCO101 vendido a 1863546-UY.
Costo: 60-EFECTIVO. Puntos: 7.

Cancelación de Reserva – cancela *idLinea fecha pasaporteCliente nacionalidadCliente clase asiento*

Descripción: Se cancela la reserva de un pasaje.

Asumir

- Los parámetros tienen los formatos descritos en los comandos anteriores.

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) El cliente con pasaporte *pasaporteCliente* y nacionalidad *nacionalidadCliente* existe en el sistema.
e) El cliente con pasaporte *pasaporteCliente* y nacionalidad *nacionalidadCliente* tiene reservado el pasaje *asiento-clase-fecha-idLinea*.

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: Cliente *pasaporteCliente-nacionalidadCliente* no ingresado.
e) ERROR: Cliente *pasaporteCliente-nacionalidadCliente* no tiene la reserva *asiento-clase-fecha-idLinea*.

Acción

- Se cancela la reserva, liberando el asiento para ser comprado o reservado.
- Se imprime en pantalla:

Reserva *asiento-clase-fecha-idLinea-pasaporteCliente-nacionalidadCliente* cancelada.

Ejemplo

> cancela INCO237 20100813 4533567 IT PRIMERA 12
Reserva 12-PRIMERA-20100813-INCO237-4533567-IT cancelada.

Avanzar Fecha de Sistema - siguienteDia

Descripción: Pasa la fecha actual del sistema al siguiente día.

Asumir

- Al calcular el siguiente día no se tomarán en cuenta los años bisiestos.

Control

f) La fecha del sistema debe estar inicializada

Mensaje de error

f) ERROR: Fecha no inicializada.

Acción

- Se le suma un día a la fecha actual.
- Se aplica la regla 3 de la aerolínea. En caso de aplicar multas a clientes se modifican sus historiales, cambiando el estado del movimiento original de RESERVA a MULTA.
- Se imprime en pantalla:
  Fecha actual *fecha*. Reservas liberadas: *n*.

  - *fecha* tiene el formato descrito en el comando iniciar.
  -*n* es la cantidad de reservas liberadas (y multas aplicadas) por la regla 3.

Ejemplo

> siguienteDia
Fecha actual 20100425. Reservas liberadas: 1.

3.1.3 Comandos de Información

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 los comandos anteriores.

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-cantAsientosPrimera -cantAsientosTurista-descripcionAvion-aeroOrigen-aeroDestino-horaSalida-duracion-frecuencia* .

Ejemplo

> infoLinea INCO100
Linea: INCO100-0-10-100-AIRBUSA300-MVD-EZE-1400-45-DIARIA.
> infoLinea INCO237
Linea: INCO237-1-20-250-BOEING747-MVD-MAD-2030-720-SEMANAL VIERNES.

Información del Cliente – infoCliente *pasaporteCliente nacionalidadCliente*

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

Asumir

- Los parámetros tienen los formatos descritos en los comandos anteriores.

Control

a) El cliente con pasaporte *pasaporteCliente* y nacionalidad *nacionalidadCliente* existe en el sistema.

Mensaje de error

a) ERROR: Cliente *pasaporteCliente-nacionalidadCliente* no ingresado.

Acción

- Se imprime en pantalla:
  Cliente: *nombre-apellido-pasaporte-nacionalidad-puntos*.
  Movimientos:
  asiento1-clase1-fecha1-idLinea1-estado1
  ...
  asientoN-claseN-fechaN-idLineaN-estadoN

  - los movimientos están ordenados de acuerdo a como fueron ingresados al sistema (primera reserva o compra va primero)
  - el *estado* puede ser COMPRA, RESERVA, CANCELA o MULTA.

Ejemplo

> infoCliente 1863456 UY
Cliente: 'Roberto'-'Gomez'-1863456-UY-7.
Movimientos:

3-TURISTA-20100427-INCO101-MULTA

1-TURISTA-20100427-INCO101-COMPRA

1-TURISTA-20100426-INCO100-COMPRA

> infoCliente 4533567 IT

Cliente: 'Patricia'-'della Giovampaola'-4533567-IT-0.

Movimientos:

12-PRIMERA-20100813-INCO237-CANCELA

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 los comandos anteriores.

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:

  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: L – libre, R – reservado y C – comprado.
  - 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.

Ejemplo

> infoVuelo INCO101 20100426
Primera:
LLLLLLLLLL
Turista:
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL

> infoVuelo INCO101 20100427
Primera:
LLLLLLLLLL
Turista:
CLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL

> infoVuelo INCO237 20100813
Primera:
LLLLLLLLLL
LLLLLLLLLL
Turista:
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL

LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL
LLLLLLLLLL

Listar Clientes – listarClientes

Descripción: Se listan en pantalla los clientes de la aerolínea.

Acción

- Se imprime en pantalla:
  Clientes:
  pasaporte1-nacionalidad1
  ...
  pasaporteN-nacionalidadN

  - los clientes están ordenados de acuerdo a como fueron ingresados al sistema.

Ejemplo

> listarClientes
Clientes:

1863456-UY

4533567-IT

Listar Líneas – listarLineas

Descripción: Se listan en pantalla las líneas de la aerolínea.

Acción

- Se imprime en pantalla:

  Lineas:

  idLinea1

  ...

  idLineaN

  - las líneas están ordenadas alfabéticamente por su identificador.

Ejemplo

> listarLineas

Lineas:

INCO100

INCO101

INCO237

Listar Vuelos que Salen Hoy – listarVuelosHoy

Descripción: Se lista en pantalla información sobre los vuelos que salen en la fecha actual. Se consideran sólo los vuelos que tienen al menos un asiento comprado.

Control

a) El comando iniciar se había ejecutado previamente.

Mensaje de error

a) ERROR: Fecha no inicializada.

Acción

- Se imprime en pantalla:

  Planilla de Vuelos que Salen Hoy:

  Linea: *idLinea1-idAvion1-cantAsientosPrimera1 -cantAsientosTurista1-descripcionAvion1-aeroOrigen1-aeroDestino1-horaSalida1-duracion1-frecuencia1*.

  Primera:

  Asiento: *p1* – Cliente: *nombreP1-apellidoP1-pasaporteP1-nacionalidadP1-puntosP1*.

  ...

Asiento: *pN* – Cliente: *nombrePN-apellidoPN-pasaportePN-nacionalidadPN-puntosPN*.

Turista:

Asiento: *t1* – Cliente: *nombreT1-apellidoT1-pasaporteT1-nacionalidadT1-puntosT1*.

...

Asiento: *tM* – Cliente: *nombreTM-apellidoTM-pasaporteTM-nacionalidadTM-puntosTM*.

...

- los vuelos están ordenados por hora de salida.

Ejemplo

```
> siguienteDia
Fecha actual 20100426. Reservas liberadas: 0.
> listarVuelosHoy
Planilla de Vuelos que Salen Hoy:
Linea: INCO100-0-10-100-AIRBUSA300-MVD-EZE-1400-45-DIARIA.
Primera:
Turista:
Asiento: 1 - Cliente: 'Roberto'-'Gomez'-1863456-UY.
```

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 ImpresiónResultado

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

En la página del curso sección Laboratorio se encuentra disponible un gráfico que muestra la dependencia entre los módulos: *dependencias.pdf*.

Cabe destacar que se aplica la propiedad transitiva, por ejemplo: el módulo *MovimientoCliente*, necesita del módulo *Linea* y *Linea* necesita de *Avion*, aplicando la propiedad transitiva, *MovimientoCliente* necesita de *Avion* también. Para simplificar el gráfico se han eliminado las flechas que se obtienen por transitividad.

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, Cliente, Línea, MovimientoCliente, VuelosHoy
2	Fecha, DicAviones
3	Historial, ListaVuelos, ListaLineas, ListaClientes
6	VuelosFuturos
7	AeroProg2
8	Vuelo
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 2

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

- 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

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 26 de Abril 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 1. 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).

1 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. 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***
altaCliente	ERROR: Existe un cliente con pasaporte *pasaporte* y nacionalidad *nacionalidad*.	`ERROR_CLIENTE_EXISTENTE`
	Cliente *pasaporte – nacionalidad* ingresado.	`ALTA_CLIENTE`
altaAvion	Avion *n* ingresado.	`ALTA_AVION`

altaLinea	ERROR: Existe una linea con identificador *idLinea*.	ERROR_LINEA_NO_EXISTE
	ERROR: No existe un avion con identificador *idavion*.	ERROR_AVION_NO_EXISTE
	Linea *idLinea* ingresada.	ALTA_LINEA
iniciar	ERROR: Fecha ya inicializada.	ERROR_FECHA_YA_INICIALIZADA
	Fecha *fecha* inicializada.	INICIAR
reserva	ERROR: Fecha no inicializada.	ERROR_FECHA_NO_INICIALIZADA
	ERROR: No existe una linea con identificador *idLinea*.	ERROR_LINEA_NO_EXISTE
	ERROR: No existe vuelo *fecha -idLinea*.	ERROR_LINEA_FECHA
	ERROR: Cliente *pasaporteCliente-nacionalidadCliente* no ingresado.	ERROR_CLIENTE_NO_EXISTE
	ERROR: Asiento *asiento-clase-idLinea* invalido.	ERROR_ASIENTO_NO_EXISTE
	ERROR: Asiento *asiento-clase-fecha-idLinea* no disponible.	ERROR_ASIENTO_NO_DISPONIBLE
	ERROR: No hay asientos disponibles en *clase-fecha-idLinea.*	ERROR_NO_HAY_ASIENTOS_DISPONIBLES
	ERROR: Comando invalido para la fecha *fecha*	ERROR_FECHA_TARDIA
	Asiento *num-clase-fecha-idLinea* reservado a *pasaporteCliente-nacionalidadCliente.*	RESERVAR
Compra	ERROR: Fecha no inicializada.	ERROR_FECHA_NO_INICIALIZADA
	ERROR: No existe una linea con identificador *idLinea*.	ERROR_LINEA_NO_EXISTE
	ERROR: No existe vuelo *fecha -idLinea*.	ERROR_LINEA_FECHA
	ERROR: Cliente *pasaporteCliente-nacionalidadCliente* no ingresado.	ERROR_CLIENTE_NO_EXISTE
	ERROR: Asiento *asiento-clase-idLinea* inválido.	ERROR_ASIENTO_NO_EXISTE
	ERROR: Asiento *asiento-clase-fecha-idLinea* no disponible.	ERROR_ASIENTO_NO_DISPONIBLE
	ERROR: No hay asientos disponibles en *clase-fecha-idLinea.*	ERROR_NO_HAY_ASIENTOS_DISPONIBLES
	ERROR: Comando invalido para la fecha *fecha*.	ERROR_FECHA_TARDIA
	ERROR: Saldo de puntos del cliente *pasaporteCliente-*	ERROR_PUNTOS_INSUFICIENTES_

	nacionalidadCliente es insuficiente.	COMPRA
	Asiento *num-clase-fecha-idLinea* vendido a *pasaporteCliente-nacionalidadCliente.* Costo: *costo-modoPago.* Puntos: *puntos.*	COMPRAR
cancela	ERROR: Fecha no inicializada.	ERROR_FECHA_NO_INICIALIZADA
	ERROR: No existe una linea con identificador *idLinea*.	ERROR_LINEA_NO_EXISTE
	ERROR: No existe vuelo *fecha -idLinea*.	ERROR_LINEA_FECHA
	ERROR: Cliente *pasaporteCliente-nacionalidadCliente* no ingresado.	ERROR_CLIENTE_NO_EXISTE
	ERROR: Cliente *pasaporteCliente-nacionalidadCliente* no tiene la reserva asiento-clase-fecha-idLinea.	ERROR_ASIENTO_NO_RESERVADO_POR_CLIENTE
	Reserva *asiento-clase-fecha-idLinea-pasaporteCliente-nacionalidadCliente* cancelada.	CANCELAR
siguienteDia	ERROR: Fecha no inicializada.	ERROR_FECHA_NO_INICIALIZADA
	Fecha actual *fecha*. Reservas liberadas: *n*.	SIGUIENTE_DIA
infoLinea	ERROR: No existe una linea con identificador *idLinea*.	ERROR_LINEA_NO_EXISTE
infoCliente	ERROR: Cliente *pasaporteCliente-nacionalidadCliente* no ingresado.	ALTA_CLIENTE
infoVuelo	ERROR: Fecha no inicializada.	ERROR_FECHA_NO_INICIALIZADA
	ERROR: No existe una linea con identificador *idLinea.*	ERROR_LINEA_NO_EXISTE
	ERROR: No existe vuelo *fecha -idLinea*.	ERROR_LINEA_FECHA
	ERROR: Comando invalido para la fecha *fecha*.	ERROR_LINEA_FECHA
ListarVuelosHoy	ERROR: Fecha no inicializada.	ERROR_FECHA_NO_INICIALIZADA