

Apellido y Nombre: _____

Legajo: _____

Comisión: *Revisante*

Ejercicio N° 1 *50/60*

Un importante banco del país cuenta con la información de todos sus clientes en un archivo secuencial con el siguiente formato:

CLIENTES (ordenado por id_sucursal, id_cliente)

id_sucursal | id_cliente | nombre y apellido | saldo a la fecha | fecha alta | fecha baja

Al final de cada mes, corre un proceso de actualización con todos los movimientos generados por cada cliente durante el mes. La información se encuentra en un archivo secuencial con la siguiente estructura:

MOVIMIENTOS (ordenado por id_sucursal, id_cliente, cod_movimiento)

id_sucursal | id_cliente | cod_movimiento (0..99) | nombre y apellido | fecha_movimiento | monto | detalle | categoría (1..6) | tipo

Donde:

- cod_movimiento indica: 0 (alta de un nuevo cliente), 99 (baja de un cliente), y cualquier otro valor entre 1 y 98 es una transacción en la cuenta del cliente.
- detalle: indica una descripción del movimiento.
- categoría: indica la categoría del movimiento (1-Supermercado, 2-Farmacia, 3-Carnicería, 4-Transferencia, 5-Pago de servicios, 6-Otros).
- tipo: indica "I" si es un ingreso, "E" si es un egreso.

Se pide:

- Desarrollar un algoritmo que permita mantener actualizado el archivo CLIENTES con sus respectivos saldos. Informar por pantalla cualquier tipo de error que considere pertinente durante el proceso.
- Indicar la cantidad de clientes que se dieron de baja el último mes.

Ejercicio N° 2 *20/40*

cantidad que se dio de baja el último mes
El banco ha solicitado un informe para conocer cómo se integra su cartera de clientes, agrupándolos por sucursal y categoría de cliente, a partir de los datos del archivo de CLIENTES (usar las estructuras de datos del ejercicio anterior).

Considerando que son 15 sucursales, y la información de las sucursales se encuentra en un archivo indexado:

Sucursales (indexado por id_sucursal)

id sucursal | nombre de la sucursal | dirección | localidad

A [10; 4]

La estructura del informe solicitado es la siguiente:

	Categoría diamante	Categoría oro	Categoría estándar	Totales x suc
Nombre sucursal 1				
Nombre sucursal 2				
...				
Nombre sucursal 15				
Totales x categoría				

La categoría del cliente dependerá de los montos obtenidos en su saldo de la cuenta al último día del mes.

- Montos menores a \$100.000 serán de categoría estándar.
- Montos menores a \$1.500.000 serán de categoría oro.
- Montos superiores serán de categoría diamante.

Nota: considerar SOLO clientes dados de baja, ya que el banco propone realizar un nuevo plan de fidelización.

Accion Egl es Ambiente

```

clientes = registro
|
| clave = registro
| | dsucursal : N(10)
| | idcliente : N(10)
| +r
| nombreApellido : AN(30)
| saldo : N(10)
| fechaAlta : fecha
| fechaBaja : fecha

```

```

Ir
|
| fecha = registro
| | dia : N(2)
| | mes : N(2)
| | ano : N(4)
| +r

```

archMae: archivo de clientes
ordenado por clave
mae: clientes

```

Movimientos = registro
|
| clave = registro
| | dsucursal : N(10)
| | idcliente : N(10)
| | codMovimiento : N(2)
| +r
| nombreApellido : AN(30)
| fechaMov : fecha
| monto : N(10)
| detalle : AN(20)
| categoria : entero
| tipo : "I", "E"

```

archMov: archivo de movimiento
ordenado por clave
mov: movimiento

Procedimiento LeerMae() es

```

| Leer(archMae, mae)
| si FDA(archMae) entonces
| | mae.clave := HV
| +r

```

Fp

Procedimiento LeerMov() es

```

| Leer(archMov, mov)
| si FDA(archMov) entonces
| | mov.clave := HV
| +r

```

Fp

```

auxiliar = registro
|
| clave = registro
| | dsucursal : N(10)
| | idcliente : N(10)
| +r
| nombreApellido : AN(30)
| saldo : N(10)
| fechaAlta : fecha
| fechaBaja : fecha

```

archAux: archivo de auxiliar ordena-
do por clave
aux: auxiliar
clientesBaja : entero

Procedimiento Igualar() es

Si mov.clave.codmovimiento = 0
entonces // alta

| Esc("Error, alta")

Sino

aux := mov

aux := mov

aux := mov

aux := mov

aux := mov

aux := mov

Procedimiento Modificar() es

Si mov.clave.codmovimiento = 0

| Esc("Error, alta invalida")

Sino

Si mov.clave.codmovimiento = 99

aux. fechaBaja := mov.

fechaBaja

clientesBaja := clientesBaja

+ 1

Sino // 1...98

Si mov.tipo = "I" entonces

aux.saldo := mov.monto +

aux.saldo

Sino

// mov.tipo "E"

aux.saldo := aux.saldo -

mov.monto

Fin

Fin

Fin

Fin

Algoritmo

Abrir E/(archMae) ; Abrir E/(archMov) ; Abrir /s (archAux)

dientesBaja := 0 ; LeerMae() ; LeerMov()

Mientras (mae.clave < HV) o (mov.clave < HV) [✓] hacer

Si (mae.clave < mov.clave) entonces

aux := mae ; Grabar(archAux, aux) ; LeerMae() [✓]

Sino

Si (mae.clave = mov.clave) entonces

aux := mae [✓]

Mientras aux.clave = mov.clave [✓] hacer

Modificar() ; LeerMov() [✓]

Fin

Grabar(archAux, aux) ; LeerMae() [✓]

Sino

Si (mae.clave > mov.clave) entonces

Si mov.clave.codMovimiento = 0 entonces // Alta [✓]

aux.clave.idSucursal := mov.clave.idSucursal

aux.clave.idCliente := mov.clave.idCliente

aux.nombreApellido := mov.nombreApellido

aux.techaAlta := mov.techaMov [✓]

aux.saldo := 0

Sino Si mov.clave.codMovimiento = 99 entonces // Baja

Esc("Error, baja inválida, cliente inexistente")

Sino // modificación

// 1 al 98 [✓]

Esc("Error, modificación inválida")

Fin

Fin

Mientras aux.clave = mov.clave entonces

Modificar() ; LeerMov() [✓]

Fin

Grabar(archAux, aux) [✓]

Fin

bien!
50/60

F_s
 F_m

Esc("La cantidad de dientes que se dieron de baja es de:", dientesBaja
".")

F_a

Acción Ey2 es
Ambiente

```
// Archivo clientes, 1
sucursales = registro
| idSucursal: N(2)
| nombreSucursal: AN(30)
| dirección: AN(30)
| localidad: AN(30)
```

↓

archSucursal: archivo de sucursal indexado por idSucursal

suc: sucursal

A: arreglo de $[1...16; 1...4]$ de entero

i, j, categoria: entero

Función categoria(x: entero): entero es

Segun x hacer

≤ 1 : $< 100,000$: categoria := 1

$> 100,000$: $> 100,000 \vee < 1,500,000$: categoria := 2

$> 1,500,000$: categoria := 3

↑_s

↑_p

Procedimiento Acero() es

Para i=1 hasta 16 hacer

Para j=1 hasta 4 hacer

$A[i, j] := 0$

↑_p

↑_p

// Archivo clientes declarado al reverso →

Cientes → registro

clave = registro

idSucursal: N(10)

idCliente: N(10)

tr

nombreApellido: AN(30)

saldo: N(20)

fechaAlta: fecha

fechaBaja: fecha

tr

fecha = registro

dia: N(2)

mes: N(2)

año: N(4)

tr

Definir antes

archMae: ~~X~~ archivo de dientes ordenado por clave

mae: dientes

Procedimiento Mostrar() es

Esc("1. Datos | Categorias | Cat 2 | Cat 3 | Total |")

Para i = 1 hasta 15 hacer

Esc("Sucursal", A[i, 1], " ", A[i, 2], " ", A[i, 3], " ", A[i, 4], " ")

Fin Para

Mostrar el nombre

Esc("Total", A[16, 1], " ", A[16, 2], " ", A[16, 3], " ", A[16, 4], " ")

Fin Para

No sigue el formato de informe pedido

Algoritmo

Abstr E/(archswisal); Abstr E/(archmae); Leer (archmae, mae)
Cero()

Mientras NFDA(archmae) entonces

Si mae.fechaBaja < " " entonces

categoria(mae.saldo) *Mal llamada función*

j := categoria(mae.saldo)

suc.idswisal := mae.idswisal

Leer(archswisal, suc)

Si EXISTE entonces

i := suc.nombreswisal

A[i, j] := A[i, j] + mae.saldo

A[i, 4] := A[i, 4] + mae.saldo

A[6, j] := A[6, j] + mae.saldo

A[6, 4] := A[6, 4] + mae.saldo

Sino

Esc("Error, swisal no encontrada")

Js

Js

Leer(archmae, mae)

Fm

Mostrar()

Fa

20/40