UTN.BA Trabajo Práctico # 1

Temas :

AyED

Metodología-Datos Primitivos-Estructuras de Control de Programas Módulos-Estructuras de Datos: Registros-Arreglos-Archivo de Texto.

Curso: K1021 Dia: Miercoles Turno: Mañana

Fecha 1er.Entrega: 0 6 / 0 8 / 2025

Fecha 2da. Entrega: _ _ / _ _ / _ _

Fecha 3er. Entrega: _ _ / _ _ / _ _ _ _

Supermercado Kotto

Nro.Leg.:	Apellido, Nombres	e-mail
232.958-0	Dominguez ,Joaquin Ezequiel	jodominguez@frba.utn.edu.ar
233.220-6	Holm ,Federico	fholm@frba.utn.edu.ar
232.984-0	Incutti ,Mateo	mincutti@frba.utn.edu.ar
233.215-2	Baquero ,Francisco	fbaquero@frba.utn.edu.ar
232.921-9	Almada ,Tomas	toalmada@frba.utn.edu.ar
233.029-5	Soria ,Francisco	Frsoria@frba.utn.edu.ar
232.930-0	Barcala Roca ,Santiago	sbarcalaroca@frba.utn.edu.ar
232.992-0	Mampaso Romero ,Brayan	bmampaso@frba.utn.edu.ar
232.941-4	Cejas ,Facundo Javier	facejas@frba.utn.edu.ar

Docente: Lic. Hugo Cuello. Es condición necesaria tener aprobado en tiempo y forma el TP para promocionar la asignatura.

Observaciones: **-del docente-**

Fecha T.P. aprobado: _ _ / _ _ / _ _ **Firma:**

Presentación de entrega de T.P.: ESTRUCTURAS DE DATOS ESTÁTICAS

- Carpeta tamaño **A4** de tapa transparente y sujetador de hojas con gancho metálico, que puede estar recubierto en plástico. Entregar con el siguiente orden:

1. Carátula (presentada anteriormente).
2. Presentación del T.P. (esta misma, no modificarla).
3. Enunciado del problema, emitir el archivo .docx original.
4. Dibujos o gráficos de las estructuras de datos utilizadas y rotuladas con valores de inicialización, espacio de memoria requerida y un ejemplo.
5. Estrategia diagramada (Bloque o Programa Principal).
6. Algoritmo diagramado (módulos) indicando el pasaje de parámetros en la cabecera.
7. Listado de la codificación en C++, sangría de dos espacios y escribir hasta una **columna 80 como máximo** para que no enrolle al siguiente renglón (listado fuente usando el Code Blocks, libre de errores de compilación), numerando las filas, tomar como modelo de estilo de escritura, los listados de los programas codificados en C++ entregado por el docente.
   Atención: El código C++ de este listado debe coincidir exactamente con la versión entregada en penDrive, que se devolverá una vez copiado.
8. Además el pendrive deberá contener la Muestra de los datos y Resultados de la ejecución del programa, según muestra de datos de cada uno de los archivos, realizados por el grupo, emitidos en formato tabular generado por el proceso según indicado en el enunciado.

- El nombre del archivo fuente debe ser **TP1VnK1_ _ _G_-Apellido Nombre.cpp** Ej. de nombre para el T.P.1, **TP1V1K1023G3-Perez Juan.** Además se entregarán los archivos de datos de texto, y el archivo de salida de los listados.
- **La salida debe estar dirigida a archivo de texto en formato tabular –en columnas- y perfectamente alineado**, nada debe salir ni por pantalla ni por impresora, ni realizar pausas de espera.
- Los nombres de archivos físicos en el *modo de apertura* **no** deben tener indicado ni la unidad ni la ruta, debe ser por defecto, solo el nombre y la extensión del archivo, para el archivo de texto, **Txt**. Se deben respetar exactamente estos nombres físicos para los archivos indicados anteriormente como los indicados en el enunciado del problema a resolver. El nombre del archivo de salida debe ser Ticket.Txt conteniendo el Ticket del único Cliente y el Listado ordenado por Código de Rubro dentro de los cuales estarán los artículos correspondientes.

Observaciones :

Definir el registro de cada archivo de acuerdo a lo solicitado en las consignas del TP, con idéntico tipo, el mismo orden en que fueron mencionados. Una diferencia en orden o tipos hará que no funcione adecuadamente.

ACLARACIÓN: En los archivos de texto se van a leer/grabar los datos en forma individual y no toda la línea, por lo que, se van a utiliar tanto en la lectura como en la escritura, el formato r.c, en donde r es el nombre del registro y c indica el nombre del campo, y cada campo será del tipo de dato que se vaya a leer/grabar. Ej.: si una linea tiene 4 datos: 34 153.87 pala de punta 1
Definimos 4 campos de tipos y nombres ; short codArt float importe str20 descrip bool estado y siendo rArt el nombre de la de tipo struct y Art es el nombre lógico del archivo, podemos leer de la siguiente manera:
Art >> rArt.codArt >> rArt.importe; Art.get(rArt.descrip Art >> rArt.estado
Todas las líneas deben tener la misma longitud física. Esto es muy importante si en algún momento queremos realizar un acceso directo a una componene (línea) del archivo.

IMPORTANTE: Estas indicaciones se complementan con las consignas dadas en el enunciado del problema a resolver, bajo el título de **Observaciones, restricciones y recursos disponibles**.

Índice

UTN.BA Materia : Algoritmos y Estructura de Datos	**T.P.#1:** SuperMercado Kotto.
Docente: Lic. Hugo A. Cuello	**2do. Vto. / 3:** 03-septiembre-2025

Trabajo Práctico #1 : Metodología-DP-ECP-Módulos-ED: Registro, Arreglo y Archivo texto.

Se requiere de un proceso que realice la simulación de compra de artículos de un cliente en un supermercado. Para ello, se cuenta con los siguientes archivos de datos:

a) **Articulos.Txt**: desordenado., **máx. 10000 artículos**, conteniendo cada línea los siguientes datos:

Cód. Art. (int máx. 8 díg.)	Cod.Rubro (short 2 díg.)	Descripción Art. (str30)	Stock Actual (ushort, 4 díg.)
Precio Unitario (float 6.2)	Uni.Medida (str10)	Porc. Ofertas (short x 14)	

NOTA: el campo **Porc. Ofertas** son 7 pares indicando pos. par: tipo descuento, pos. impar: porcentaje descuento.

b) **IndDescripArt.Txt:** ordenado por Descripción de Artículos**,** conteniendo:

Descripción Art.	Posición Art. (int)	Estado (bool)

NOTA: el campo Estado valor 0 (falso) indica Cod.Art. baja lógica, valor 1 (true) indica Cod.Art. activo.

c) **Rubros.Txt**: ordenado por **Cód.Rubro**, con 15 rubros, conteniendo los siguientes datos:

Cód. Rubro	Descripción Rubro (str20)

d) **ListaCompras.Txt**: sin orden, cada una de las líneas contiene los siguientes datos:

Descripción Art.	Cant. Requerida (short 2 díg.)

Observación: En cada **struct** (registro mejor dicho) se deben indicar solamente los campos indicados.

Se pide:

1. Volcar y Generar en la memoria **RAM** estructuras estáticas (Tablas):

   1.a) Volcar a una primer tabla todos los datos del archivo **IndDescripArt.Txt**:.
   1.b) Volcar a una segunda tabla todos los datos del archivo **ListaCompras.Txt**:.
   1.c) Volcar a una tercer tabla conteniendo: *Cód.Rubro* (con repetición) y su *Pos.Art.*, luego, ordenar por el campo *Cód.Rubro*, del archivo **Articulos.Txt.** Esta tabla contiene la misma cantidad de componentes que la tabla del archivo **IndDescripArt**. del punto 1.a).

2. Procesar la **tabla de Lista de Compras** recorriendo secuencialmente y por cada *descripción* de artículo buscarlo en la **tabla de ÍndDescripArt**, si el estado es activado se deberá restar el stock actual, **actualizando** en el archivo el *stock actual*; se pueden presentar dos casos: 1) el stock Actual es mayor o igual al solicitado o 2) es menor en este caso se satisface parcialmente la solicitud comprada y se deberá indicar la cantidad efectiva comprada. Si el *estado* esta en baja lógica en la tabla correspondiente se deberá indicar el valor cero en el campo *cantidad comprada*.
3. Emitir el ticket con igual criterio que la **Lista de Compras** de acuerdo al siguiente diseño:

Datos de la Cabecera son:

K O T T O
Yo te reconozco
SUC 170
XXXXXX…X 9999
XX….X
C.U.I.T. 99-99999999-9
Fecha: *nomdia* 99/99/9999
Hora: 99:99:99

```
Nro. Ticket: 9999-99999999
Nro. Caja: 9999
-----------------------------------------
F A C T U R A - B
ORIGINAL
-----------------------------------------
```

Datos del cuerpo son, en formato tabular (en columnas):

```
  4 x $  745.32
Galletitas Media Tarde x 3    pack
99999999                              $    9999.99
Jub.          6                       $     -999.99

 5 x $  3962.04
Cuadril novillito           kgs
99999999                              $   99999.99
Marca.        5                       $   -9999.99

 3 x $  3845.12
Coca-Cola 225               litros
19967859                              $   11535.36
MercPago      6                       $   -2883.84

15 x $  643.23
Galletitas Express pack x 3   gramos
62937159                              $    9648.45

 2 x $  4908.45
Alfajor TERRABUSI 6 unidades  pack
99999999                              $    9999.99
Comunid.      6                       $   -9999.99

SubTot. sin descuentos....:           $  999999.99
Descuentos por promociones:           $  -99999.99
=======================================
T O T A L                             $   999999.99
=======================================
```

Datos del pié son:

```
Su pago con Tipo Pago:                $ 999999.99
Su vuelto:                            $      9.99
            G R A C I A S  P O R  S U  C O M P R A
              Para consultas, sugerencias o reclamos
              comunicarse al correo infoKotto.com.ar
```

NOTA: El cuerpo del ticket se compone de:

cant. x pre.Uni
Descripción del art Unid.Med.
Cód. Art. Imp.Tot.Item
Tipo descuento nroDesc. Imp.Descto

4. Emitir el Listado de Artículos ordenado por **Cód. Rubro** según el siguiente diseño:

```
-------------------------------------------------------------------------------------------------------
                    Listado de Articulos ordenados por Código de Rubro
=======================================================================
Cod. Rubro: 1 LACTEOS
Cod.Art. Descripcion                    Stk.   Pre.Uni. U.Med.  TD % TD % TD % TD % TD % TD % TD %
-------------------------------------------------------------------------------------------------------
58791254 Helados bombom                   25     802.65 gramos  1 20 1 20 1 15 4 15 3 25 2 25 5 20
74141296 Leche Serenisima con Vit.A+D     38     785.09 litros  6 20 3 20 2 15 4 15 3 25 2 25 5 20
 5978451 yogur entero c/colchon durazno   58    2041.02 litros  2 20 3 20 0 15 2 15 2  5 3 25 6 20
 3236875 Queso reggianito                 23    1873.30 kilo    4 20 3 20 1 15 3 15 3 25 3 25 3 20

Cod. Rubro: 3 ROPA
Cod.Art. Descripcion                    Stk.   Pre.Uni. U.Med.  TD % TD % TD % TD % TD % TD % TD %
-------------------------------------------------------------------------------------------------------

Cod. Rubro: 5 BEBIDAS CON ALCOHOL
Cod.Art. Descripcion                    Stk.   Pre.Uni. U.Med.  TD % TD % TD % TD % TD % TD % TD %
-------------------------------------------------------------------------------------------------------
80196873 Cerveza AmsTel Lager            250    3421.87 litros  2 20 2 20 6 15 2 15 4 25 3 25 2 20
13102456 Cerveza Schneider               250     936.05 gramos  1 20 3 20 5 15 1 15 1 25 1 55 5 20
78678241 Vino Toro Malbec                250     809.09 litros  2 20 2 20 4 15 4 15 2 25 0 25 6 20

Cod. Rubro: 8 BEBIDAS SIN ALCOHOL
Cod.Art. Descripcion                    Stk.   Pre.Uni. U.Med.  TD % TD % TD % TD % TD % TD % TD %
-------------------------------------------------------------------------------------------------------
```

```
19967859 Coca-Cola 225                  81    3845.12 litros  2 20 4 10 2 15 3 10 1  5 4 25 6 25
```

En el bloque principal, solo se establecen las invocaciones a los módulos, y declarar las variables pertenecientes a este módulo y sus tipos de datos.

El bloque principal debe contener las siguientes acciones:

```
main()  {
    Declarar las variables utilizadas en el bloque ppal().

    Abrir (Articulos,IndDescripArt,Rubros,ListaCompras);
    VolcarArchivos(lista de parámetros que correspondan); // indicados por el grupo de trabajo.
    ProcCompras(lista de parámetros que correspondan);
    EmitirTicket(lista de parámetros que correspondan);
    EmitirArt_x_Rubro(lista de parámetros que correspondan);
    Cerrar (Articulos,IndDescripArt,Rubros,ListaCompras);
    return 0;
}
```

Observaciones, restricciones y recursos disponibles:

Utilizar las siguientes funciones, se indican los prototipos, invocando en donde sea necesario:

- *bool* **Leer*Suf*** (*modo* &id, *sid* &id), el cual **lee una componente de datos** del archivo y los almacena en una estructura interna. La función retorna un boolean, verdadero, si la lectura fue exitosa, caso contrario, falso; *modo* puede ser *fstream* o *ifstream* según corresponda, *sid* indica un tipo de estructura de datos. Crear una función de lectura para cada archivo que se lee. Suf luego de Leer en el nombre de la función es un sufijo para el archivo que se lee, el cual debe ser reemplazado Suf por el nombre apropiado, p.e. *LeerArt*, *LeerRub*…
- void **CabeceraTicket**(**int** &ds) se debe invocar dentro del módulo **EmitirTicket**. El parámetro *ds* se deberá utilizar en el cuerpo del ticket, que indica el día de la semana, 1:dom., 2:lun., … 7:sáb.
- void **PieTicket**(float impTot, float impTotDesto, float impTotConDesto) se debe invocar dentro del módulo **EmitirTicket**.
- *void* **OrdxBur** (*tid* tbl, *tid* card), *ordena* tbl *Rubros en Artículos con repetición* por *descripción.*
- *void* **IntCmb** (*id* &elem1, *id* &elem2), *intercambia ambos elementos.*
- *void* **ActLinea***(modo &id, sid id)* que debe *actualizar en la línea que corresponda* el nuevo *stock actual.* Se debe grabar c/u. de los datos contenidos en la línea.
- *int* **BusBinVec***(tbl* **id**, *tid* **clv**, *tid* **ult***)* que busca *el valor clv* en la tabla de *IndDescrip* ***y que*** *retorna la posición encontrada o -1* si no se encontró.
- *string* **Replicate**(*char* car, *unsigned* n), retorna una cadena con n veces car.
- *long* **GetTime**(*int* hora,*int* min, *int* seg). *La función retorna la hora larga, como un solo número en el formato hhmmss. Además en sus parámetros devuelve la hora, los min. y los segundos*.
- *long* **GetDate***(int* year,*int* mes,*int* dia,*int* diaSem*). La función retorna la fecha larga, como un solo número en el formato aaaammdd. Además en sus parámetros devuelve el año, el mes, el día y el día de la semana.* Esta función se encuentra en OBTENER LA FECHA Y HORA DEL SISTEMA del apunte del prof. Hugo Cuello Teoría y Práctica del Lenguaje C/C++ *ANEXOS*

Espacio en disco: Solo para generar el archivos de salida **Ticket.Txt** en formato texto.
Espacio para arrays y registros: Lo necesario que requiera este proyecto.
Espacio en memoria dinámica: 0 bytes.

Accesos a los archivos: un solo recorrido secuencial, para leer en *Artículos*, *IndDescripArt* y en *ListaCompras*. Además acceso al azar para leer y/o grabar en *Articulos en 3 (tres) instancias; una para actualizar el saldo actual, dos para emitir el ticket y 3 para emitir listado de rubros*.

Bloque Principal: sólo invocaciones a módulos, según lo establecido anteriormente.

Paradigma de Programación: Solo se aceptará el Paradigma Imperativo Procedural, Programación Estructurada y Modular.

Optimización: dado que el uso de ciclos afecta el tiempo de ejecución de un proceso, se evaluará la eficiencia en el uso de los mismos.

Utilizar nombres significativos para los identificadores, dibujos para las estructuras de datos a utilizar, rotulando cada elemento, tamaño, breve leyenda de cómo se generan y estado inicial, respetar esos nombres para utilizarlos en el algoritmo. Preparar una muestra de datos para los archivos de datos e imprimirla, para ejecutar el programa. En el disco solo contendrán los archivos "**TP1V_K1_ _ _G_-Apellido Nombre.cpp**", y los archivos de datos indicados anteriormente como así también el archivo de salida todos ubicados en la carpeta raíz del disco. Ejemplo: TP1V**1**_K1**023**G**3_PEREZ JUAN**.CPP

Cada grupo debe crear su propia muestra de datos para los archivos, artículos de un supermercado.

Se deben utilizar constantes con nombres para indicar cantidades.

El Trabajo Práctico deberá ser entregado de acuerdo a las pautas indicadas más abajo, el cual se aprobará si reúne los requerimientos solicitados en **tiempo y forma (tres fechas máximas)**: A: Aprobado, N: No Aprobado. (**Se debe respetar el orden indicado a continuación**):

1. Entregar en carpeta tamaño A4 de tapa transparente y con sujetador de gancho perfectamente alineadas para las hojas lo siguiente: **(no se aceptan hojas sueltas ni otro tipo de carpeta)**
2. Carátula con los datos de los integrantes del TP, la cantidad de alumnos por grupo del mismo curso se determinará en clase no superando 5 grupos como máximo.
3. Esta misma hoja que establece el enunciado del problema a resolver. Cada grupo elegirá un líder del proyecto, que será el responsable de realizar las entregas del TP. Si un líder abandona la cursada, se deberá elegir otro líder.
4. Diseñar las estructuras de Datos graficándolas indicando con rótulos apropiados, cada elemento, su tamaño en bytes y las variables utilizadas. Las estructuras de datos a graficar son:
    a. *El diseño de los registros de cada uno de los archivos.*
    b. *Otras estructuras de datos que considere necesarias para poder realizar el proceso solicitado. Algunas serán explicadas en clase, como complemento a este documento.*
5. Graficar el Bloque Principal.
6. Graficar cada uno de los módulos –funciones- a utilizar, cabecera y cuerpo.
7. Construir una muestra de datos, para los archivos de datos, la cual se la utilizará para probar el Algoritmo. En la hoja impresa a entregar debe haber rótulos apropiados, pero, NO en los archivos de Datos, el cual contendrán solamente, los datos. Ver detalle del formato más abajo.
8. Emitir según la muestra establecida, los resultados esperados, siempre acompañada de los rótulos apropiados, según formato de salida indicados anteriormente.
9. Codificación del Algoritmo completo en el Lenguaje C++, emitiendo números de líneas. Usar Code-Blocks. Las primeras líneas serán de comentario indicando: **Nombre del programa, fecha entrega, Nro. versión, breve comentario del objetivo del programa, datos del curso, nombre del día, turno, nro. del grupo e integrantes (Apellido, Nombre). Nombre del compilador: Borland C++ V.5.5**

Formato del archivo de datos Articulos.Txt

Se deberá ajustar la escritura de los datos respetando a raja tabla este formato en donde cada columna representa un dato y su tipo de dato y ancho:

Cada columna representa de izquierda a derecha lo siguiente:

codArt	**codRub**	**descripc**	**stkAct**	**preUni**	**UniMed**	**T**	**%**	**T**	**%**	**…**	**T**	**%** (7 veces)
9(8)	99	X(30)	9(4)	9(5).99	9(10)	9	99	9	99	…	9	99

Los números enteros o reales ajustado a la derecha, las cadenas ajustadas a la izquierda. Si una cadena tiene menos caracteres de los indicados, se deberá rellenar con espacios en blanco a derecha.

Ejemplos:

```
12406297  23 Naranjas de jugo                54   1526.28 gramos     2 20 3 10 5 10 1 25 7 15 6 25 2 10
  345678 145 Leche Cindor                   2451    630.86 cc         1  5 4 15 4 15 3 20 4 10 6 25 2 15
 7451932   1 Fanta 2.25                     153   3542.09 litros     5 15 3 20 5 20 4 15 2 20 6 25 3 10
```

Para los ingresos de los datos de cada archivo de texto, utilizar el editor Code-Blocks.
Se utilizará este formato semejante a los demás archivos.

La salida de los resultados debe estar dirigida a un archivo de texto con el nombre **Ticket.Txt** se debe utilizar la sentencia **freopen**, para redirigir la salida de la pantalla a archivo de texto al utilizar **cout**.
La cantidad de datos de muestra para cada archivo deberán ser los siguientes:

Articulos.Txt: entre 40 y 45 líneas.
IndDescripArt.Txt: entre 40 y 45 líneas, con la misma cantidad que en Artículos.Txt.
Rubro.Txt: 15 líneas, si o si.
ListaCompras.Txt: entre 25 y 30 líneas.

Cada línea representa un conjunto de datos que reúne todos los datos solicitados para cada archivo.

En las muestras de datos contemplar, todos los casos posibles, con respecto a las cantidades:

- Stock insuficiente para algunos artículos.
- Sin Stock, es decir cero.
- Estado cero (*false*) algunos artículos y en Estado uno (*true*) varios.

Las opciones para las promociones son 7:
SinPromo, Promo, Marca, Jub., Comu., MercPago, ANSES, para saber si un artículo tiene o no promoción compararlo con la cadena "**SinPromo"**. Utilizar una selección múltiple para asignar el nombre de la promo a la variable. Se aclara que el tipo de promo y su pocentaje se utilizará un array de tipo *short* con 14 (catorce) posiciones en donde las posiciones pares serán los tipos de promociones y las posiciones impares indicarán el porcentaje.
Se deben leer todos los datos de cada línea en los archivos, sin importar si algunos datos no se utilicen en el proceso.
Dar nombres de identificadores representativos a su uso, es decir, con significado.

Cada nueva entrega además del nombre indicado para el archivo del código en C++ irá acompañado de la versión entregada, iniciando la primera entrega con el sufijo V1, luego la segunda entrega V2, y así sucesivamente.

(S.E. ú O.)

Diagramas

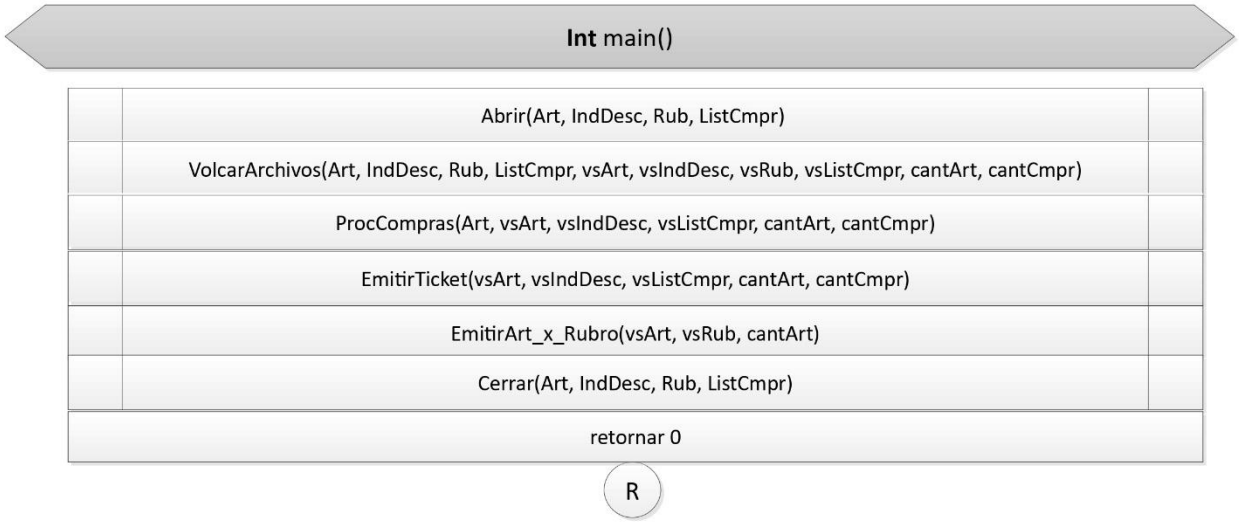
	tsRub(23B)	
	short codRub (2B)	**str20** descRub (21B)
0	1	"Frutas"
1		
2		
15		
	tsRub ***tvsRub[CANT_RUB](345B)***	

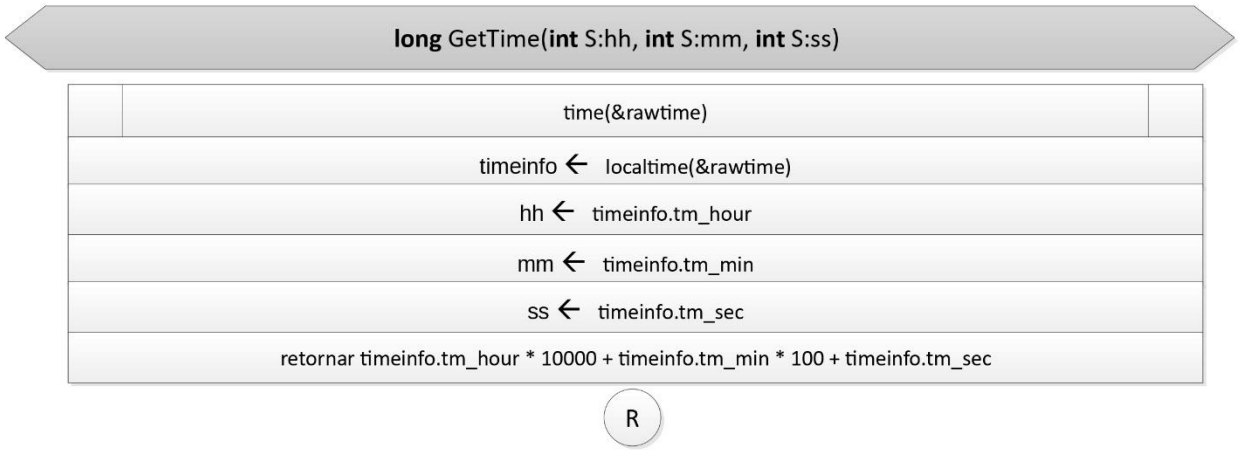
	tsIndDesc (36B)		
	str30 descArt (31B)	**int** posArt (4B)	**bool** estado (1B)
0	"Aceite de Coco"	4	1
1			
2			
10000			
	tsIndDesc ***tvsIndDesc[MAX_ART](360000B)***		

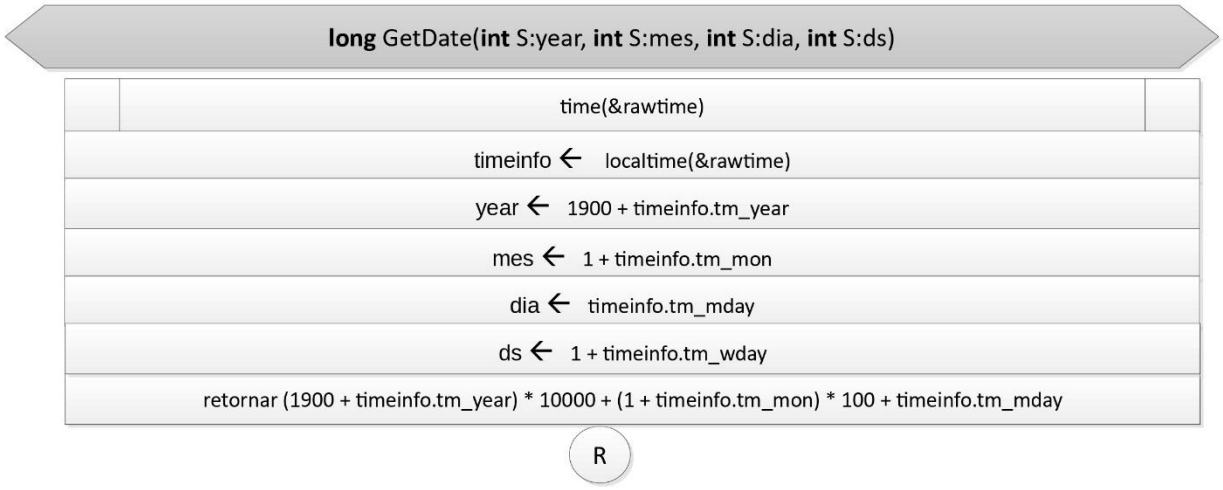
tsArt (82B)						
int codVen (4B)	**short** codRub (2B)	**str30** descArt (31B)	**ushort** stock (2B)	**float** preUni (4B)	**str10** medida (11B)	**short** ofertas[14] (28B)

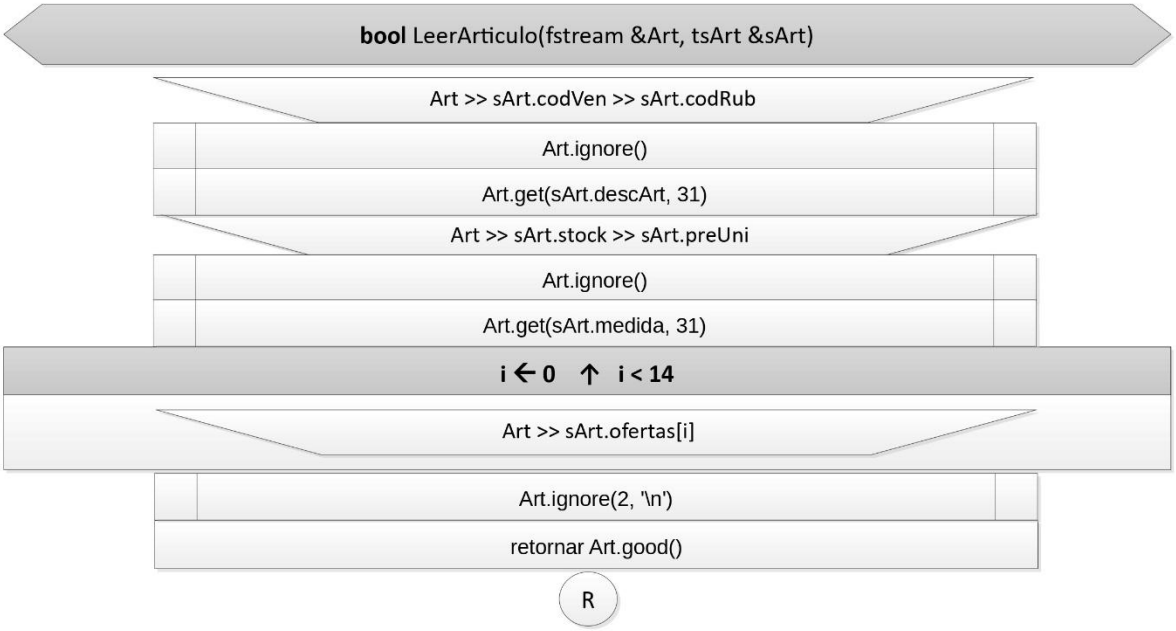
	tsArtRub(8B)	
	short codRub (2B)	**int** posArt (4B)
0	11	10
1		
2		
10000		
	tsArtRub ***tvsArtRub[MAX_ART] (80000B)***	

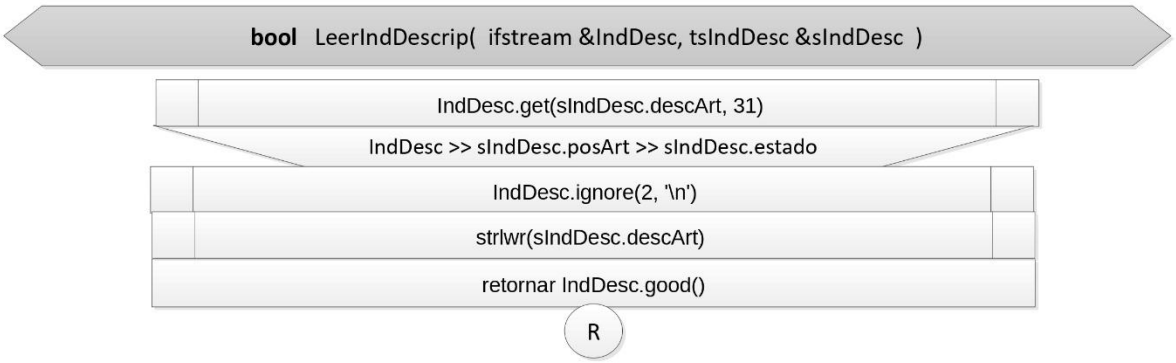
	tsCompra (33B)	
	str30 descArt (31B)	**short** cantReq (2B)
0	"Salsa de Tomate"	5
1		
2		
100		
	tsCompra ***tvsListCmpr[MAX_COMPRAS](330B)***	

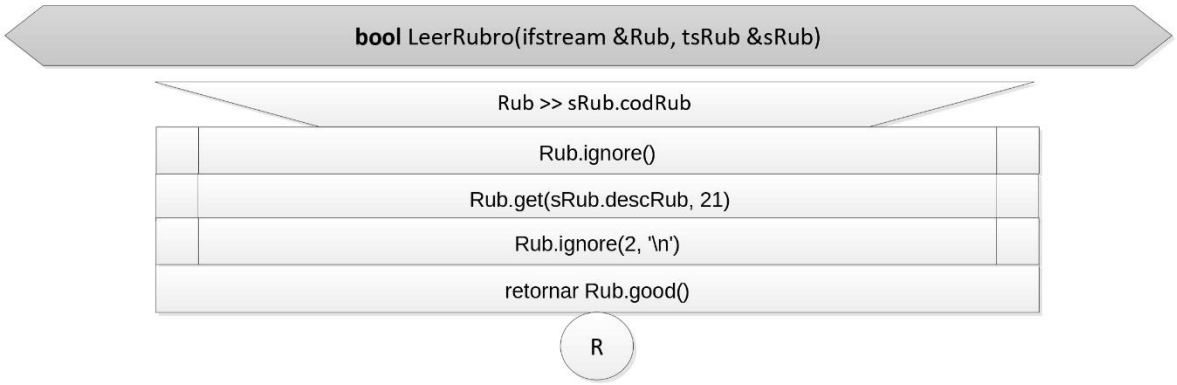

Int main()
Abrir(Art, IndDesc, Rub, ListCmpr)
VolcarArchivos(Art, IndDesc, Rub, ListCmpr, vsArt, vsIndDesc, vsRub, vsListCmpr, cantArt, cantCmpr)
ProcCompras(Art, vsArt, vsIndDesc, vsListCmpr, cantArt, cantCmpr)
EmitirTicket(vsArt, vsIndDesc, vsListCmpr, cantArt, cantCmpr)
EmitirArt_x_Rubro(vsArt, vsRub, cantArt)
Cerrar(Art, IndDesc, Rub, ListCmpr)
retornar 0
R

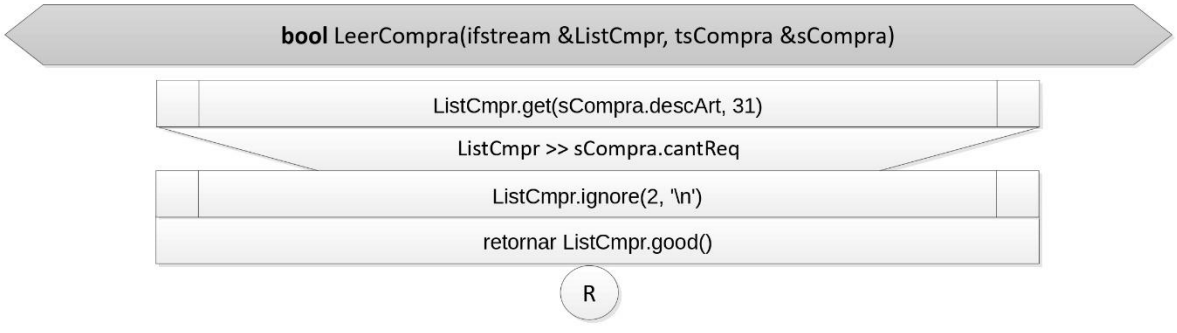

long GetTime(int S:hh, int S:mm, int S:ss)
time(&rawtime)
timeinfo ← localtime(&rawtime)
hh ← timeinfo.tm_hour
mm ← timeinfo.tm_min
ss ← timeinfo.tm_sec
retornar timeinfo.tm_hour * 10000 + timeinfo.tm_min * 100 + timeinfo.tm_sec
R

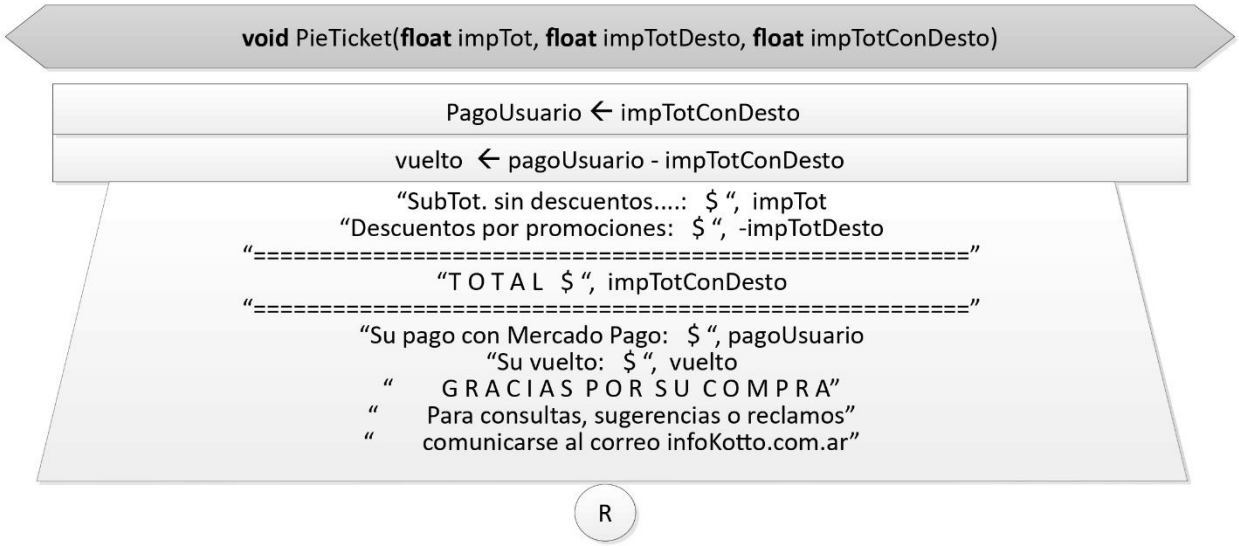

long GetDate(int S:year, int S:mes, int S:dia, int S:ds)
time(&rawtime)
timeinfo ← localtime(&rawtime)
year ← 1900 + timeinfo.tm_year
mes ← 1 + timeinfo.tm_mon
dia ← timeinfo.tm_mday
ds ← 1 + timeinfo.tm_wday
retornar (1900 + timeinfo.tm_year) * 10000 + (1 + timeinfo.tm_mon) * 100 + timeinfo.tm_mday
R

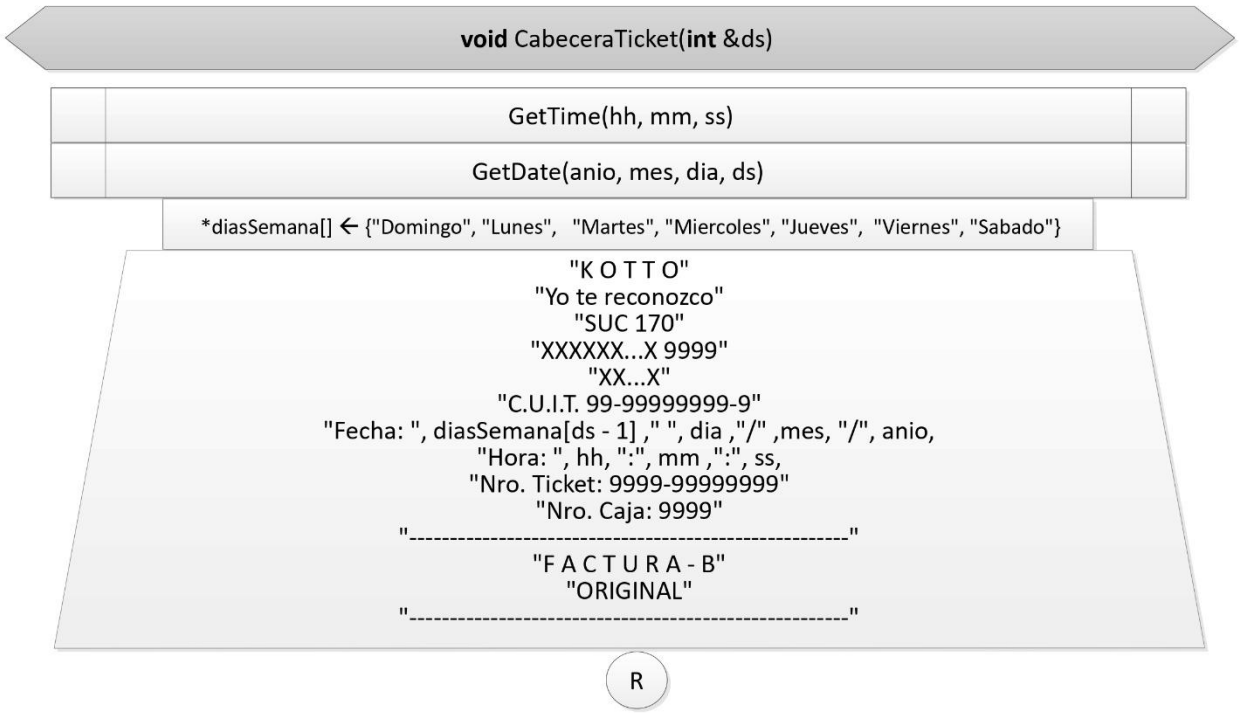

bool LeerArticulo(fstream &Art, tsArt &sArt)
Art >> sArt.codVen >> sArt.codRub
Art.ignore()
Art.get(sArt.descArt, 31)
Art >> sArt.stock >> sArt.preUni
Art.ignore()
Art.get(sArt.medida, 31)
i ← 0 ↑ i < 14
Art >> sArt.ofertas[i]
Art.ignore(2, '\n')
retornar Art.good()
R

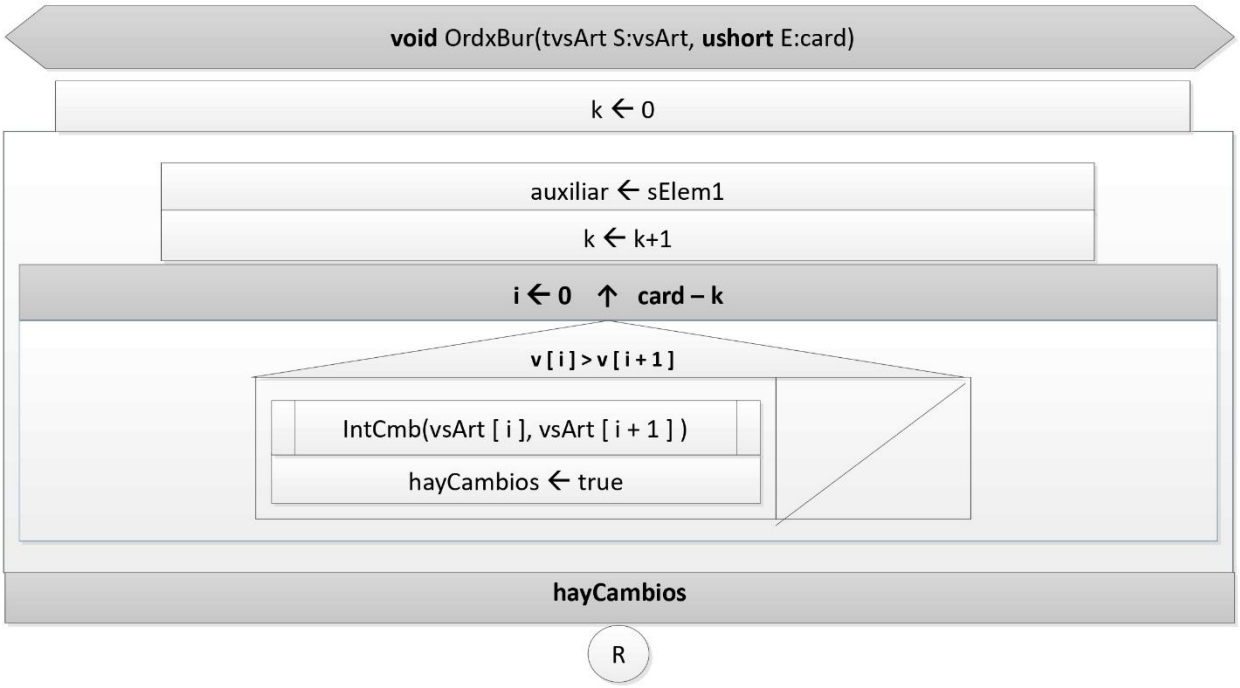

bool LeerIndDescrip(ifstream &IndDesc, tsIndDesc &sIndDesc)
IndDesc.get(sIndDesc.descArt, 31)
IndDesc >> sIndDesc.posArt >> sIndDesc.estado
IndDesc.ignore(2, '\n')
strlwr(sIndDesc.descArt)
retornar IndDesc.good()
R

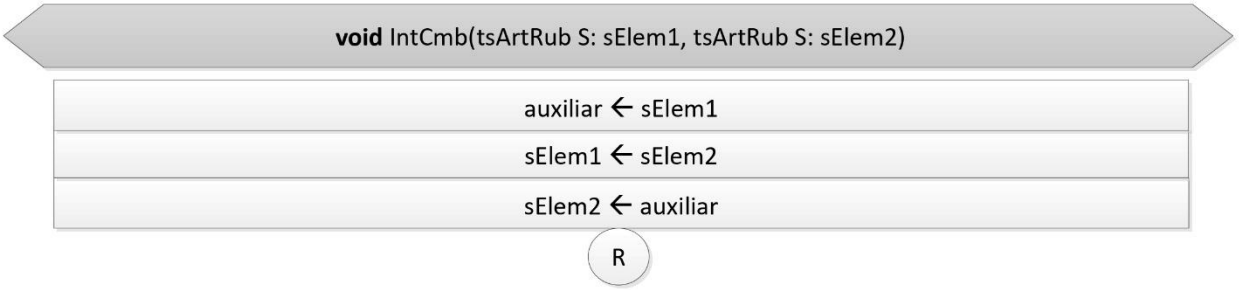

bool LeerRubro(ifstream &Rub, tsRub &sRub)
Rub >> sRub.codRub
Rub.ignore()
Rub.get(sRub.descRub, 21)
Rub.ignore(2, '\n')
retornar Rub.good()
R

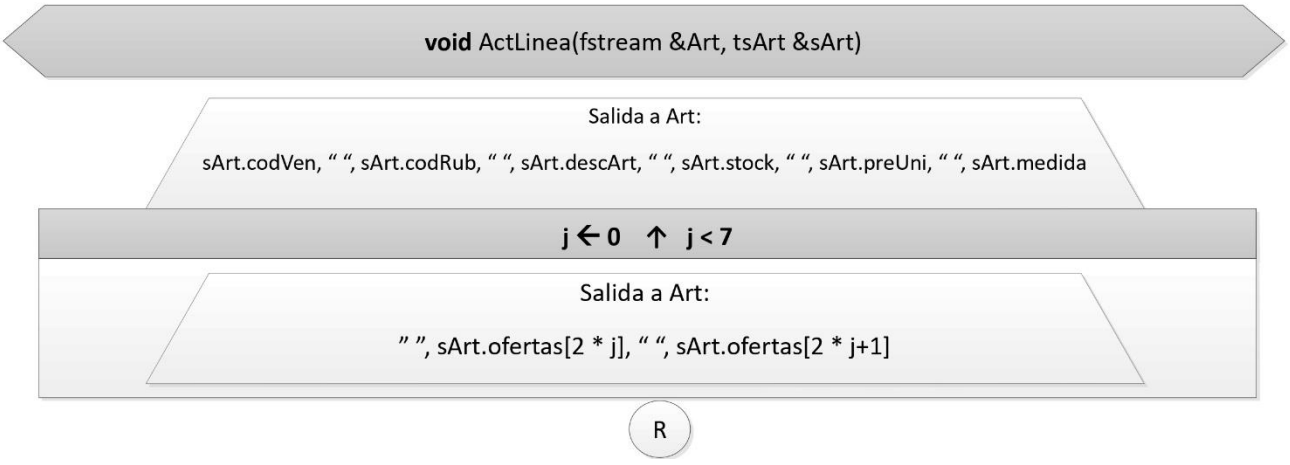

bool LeerCompra(ifstream &ListCmpr, tsCompra &sCompra)
ListCmpr.get(sCompra.descArt, 31)
ListCmpr >> sCompra.cantReq
ListCmpr.ignore(2, '\n')
retornar ListCmpr.good()
R

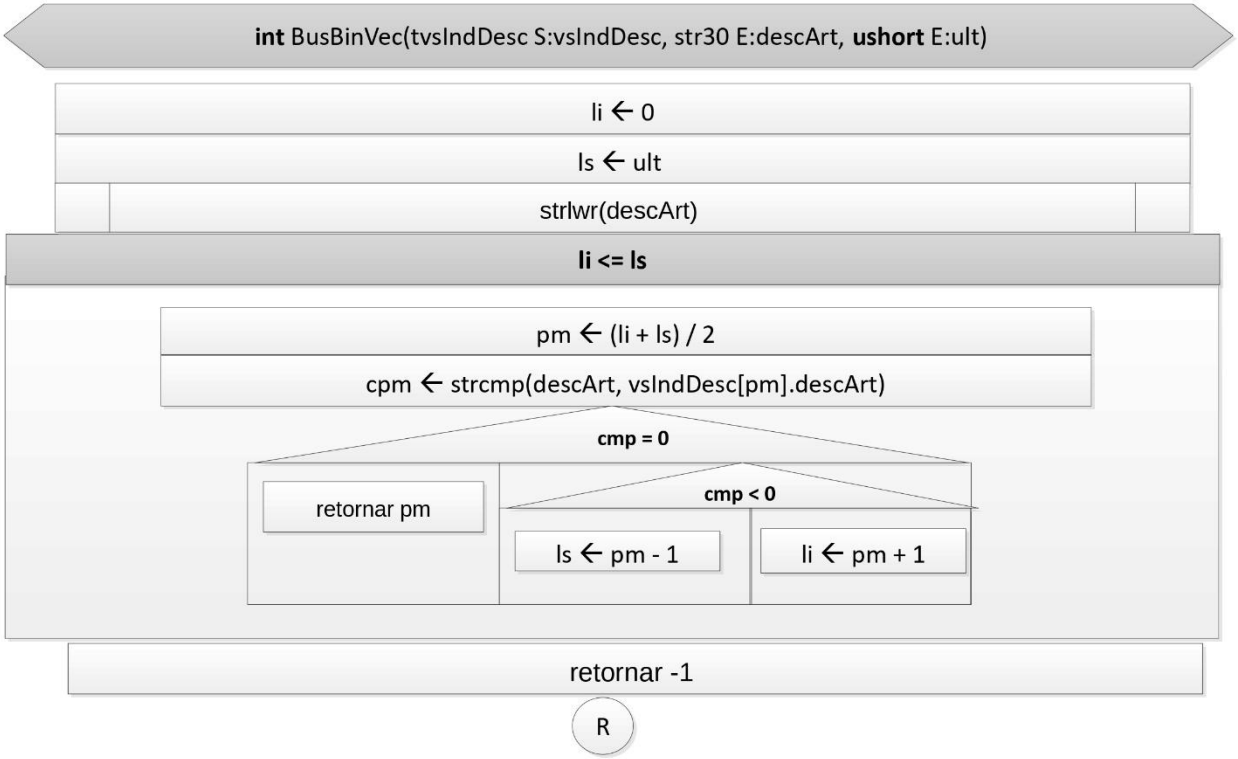

void PieTicket(float impTot, float impTotDesto, float impTotConDesto)
PagoUsuario ← impTotConDesto
vuelto ← pagoUsuario - impTotConDesto
"SubTot. sin descuentos....: $ ", impTot
"Descuentos por promociones: $ ", -impTotDesto
"=========="
"T O T A L $ ", impTotConDesto
"=========="
"Su pago con Mercado Pago: $ ", pagoUsuario
"Su vuelto: $ ", vuelto
" G R A C I A S P O R S U C O M P R A"
" Para consultas, sugerencias o reclamos"
" comunicarse al correo infoKotto.com.ar"
R

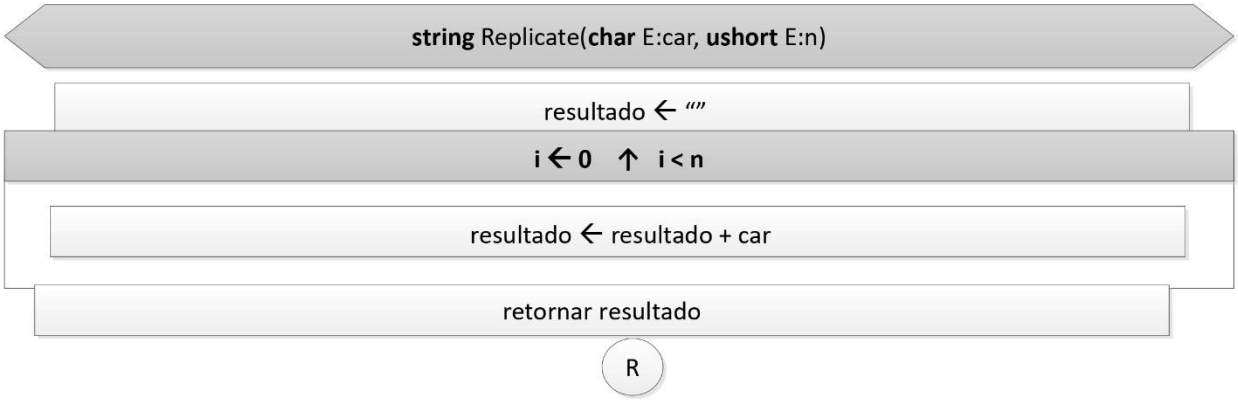

void CabeceraTicket(int &ds)
GetTime(hh, mm, ss)
GetDate(anio, mes, dia, ds)
*diasSemana[] ← {"Domingo", "Lunes", "Martes", "Miercoles", "Jueves", "Viernes", "Sabado"}
"K O T T O"
"Yo te reconozco"
"SUC 170"
"XXXXXX...X 9999"
"XX...X"
"C.U.I.T. 99-99999999-9"
"Fecha: ", diasSemana[ds - 1] ," ", dia ,"/" ,mes, "/", anio,
"Hora: ", hh, ":", mm ,":", ss,
"Nro. Ticket: 9999-99999999"
"Nro. Caja: 9999"
"----------"
"F A C T U R A - B"
"ORIGINAL"
"----------"
R

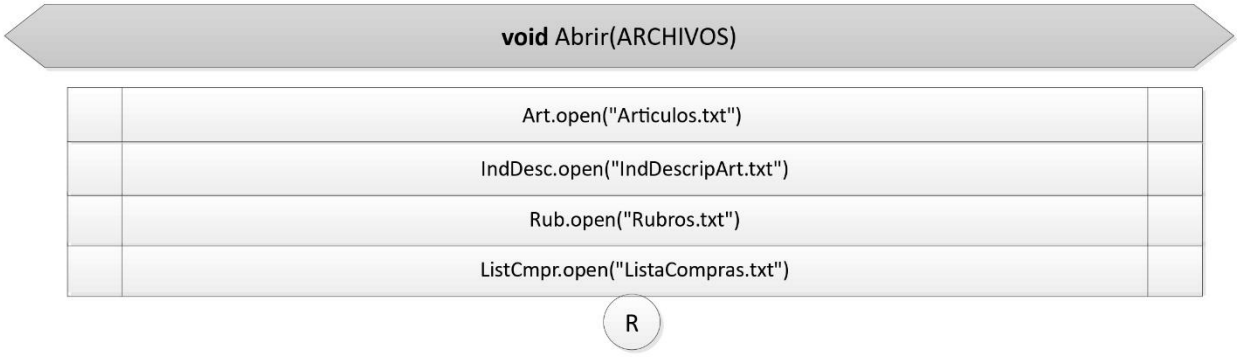

void OrdxBur(tvsArt S:vsArt, ushort E:card)
k ← 0
auxiliar ← sElem1
k ← k+1
i ← 0 ↑ card – k
v [i] > v [i + 1]
IntCmb(vsArt [i], vsArt [i + 1])
hayCambios ← true
hayCambios
R

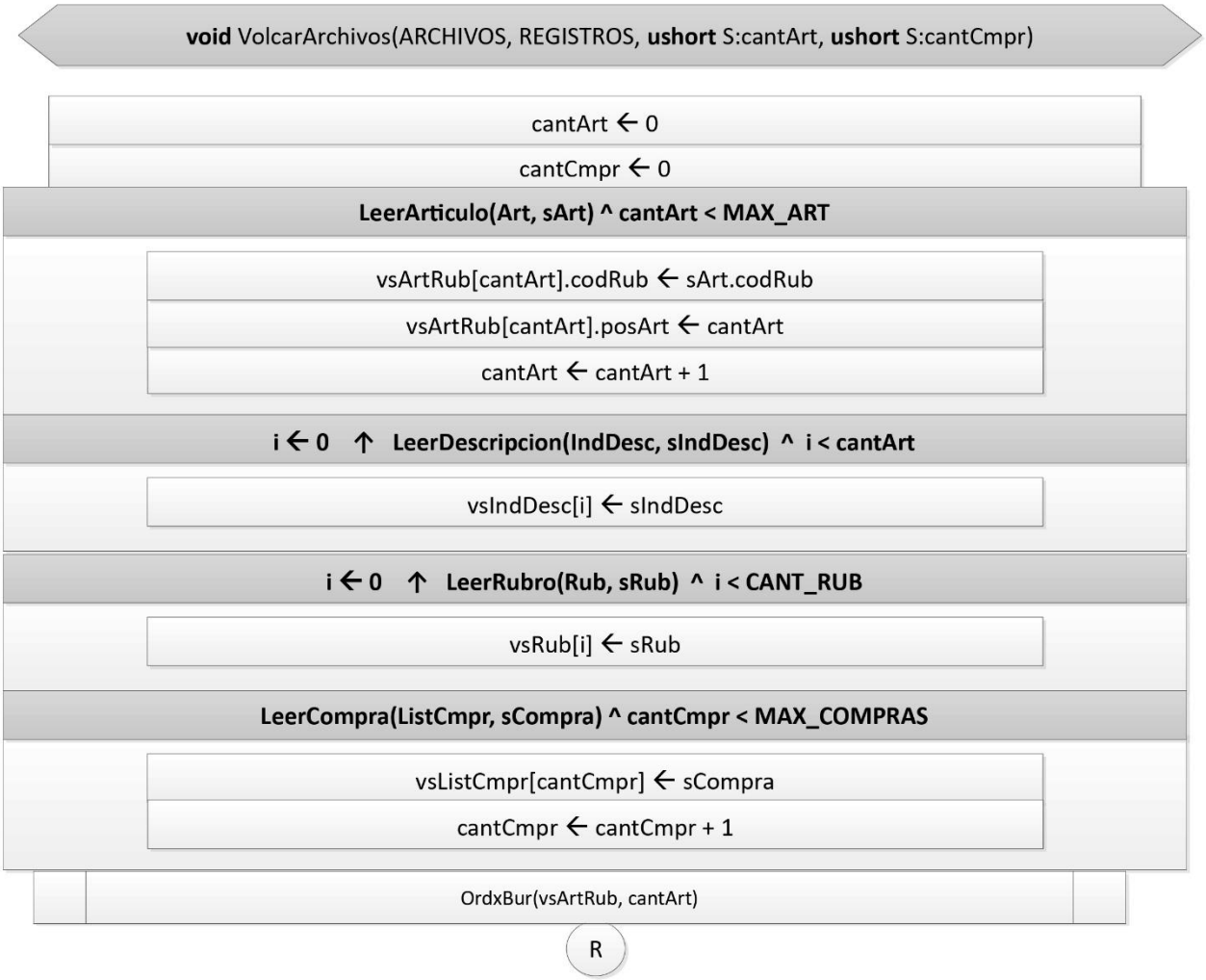

void IntCmb(tsArtRub S: sElem1, tsArtRub S: sElem2)
auxiliar ← sElem1
sElem1 ← sElem2
sElem2 ← auxiliar
R

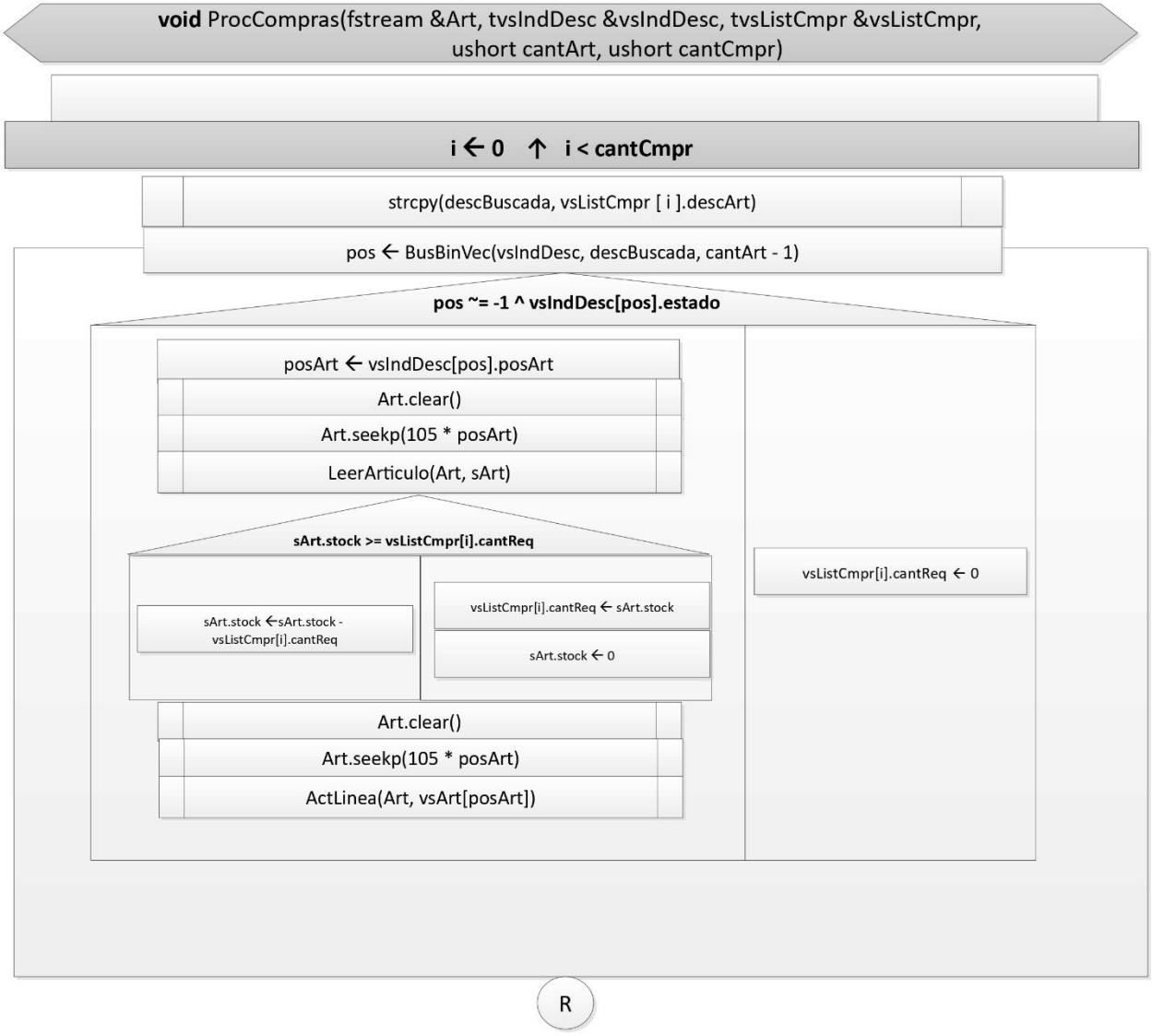

void ActLinea(fstream &Art, tsArt &sArt)
Salida a Art:
sArt.codVen, " ", sArt.codRub, " ", sArt.descArt, " ", sArt.stock, " ", sArt.preUni, " ", sArt.medida
j ← 0 ↑ j < 7
Salida a Art:
" ", sArt.ofertas[2 * j], " ", sArt.ofertas[2 * j+1]
R

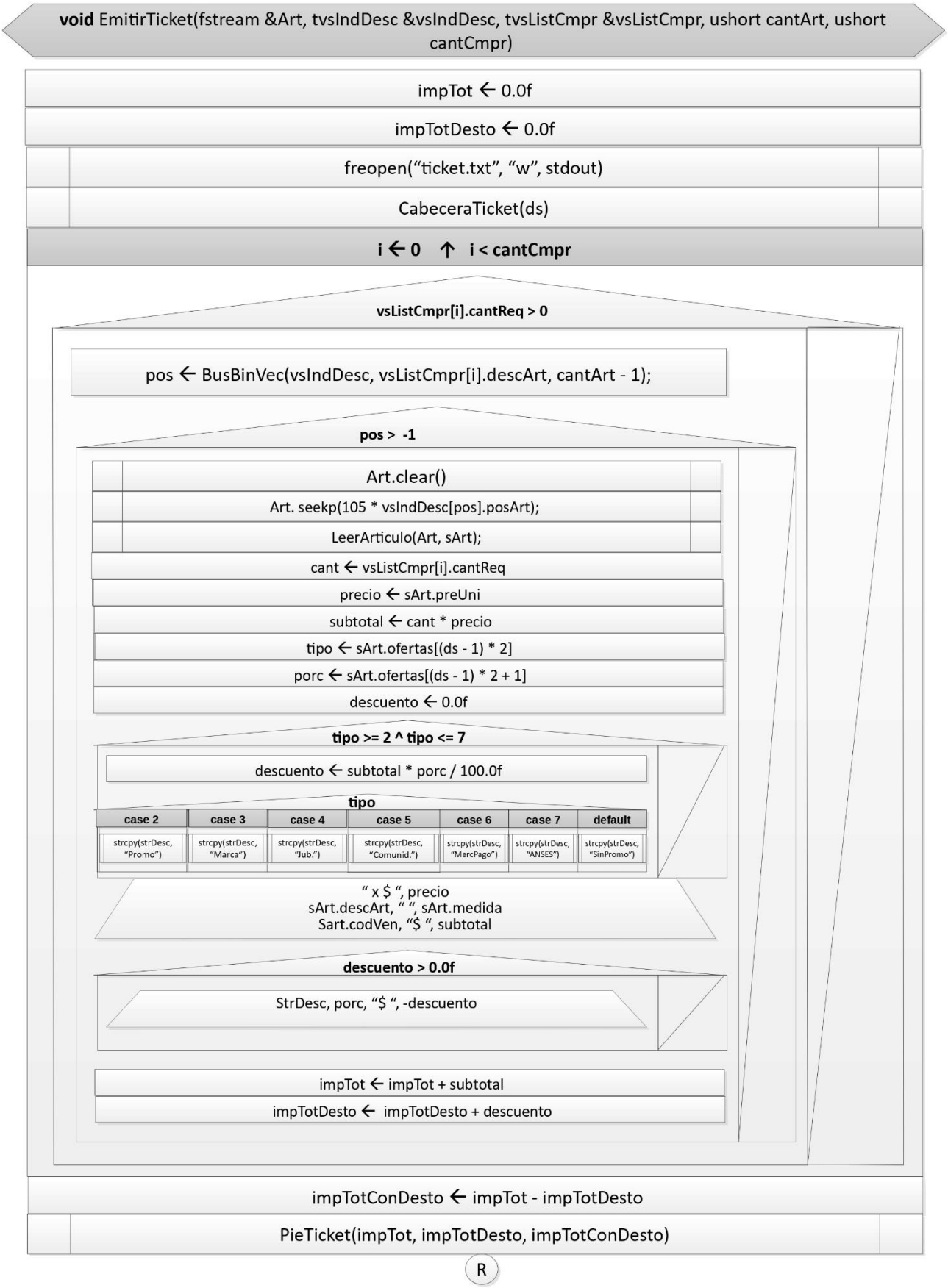

int BusBinVec(tvsIndDesc S:vsIndDesc, str30 E:descArt, ushort E:ult)
li ← 0
ls ← ult
strlwr(descArt)
li <= ls
pm ← (li + ls) / 2
cpm ← strcmp(descArt, vsIndDesc[pm].descArt)
cmp = 0
retornar pm
cmp < 0
ls ← pm - 1
li ← pm + 1
retornar -1
R

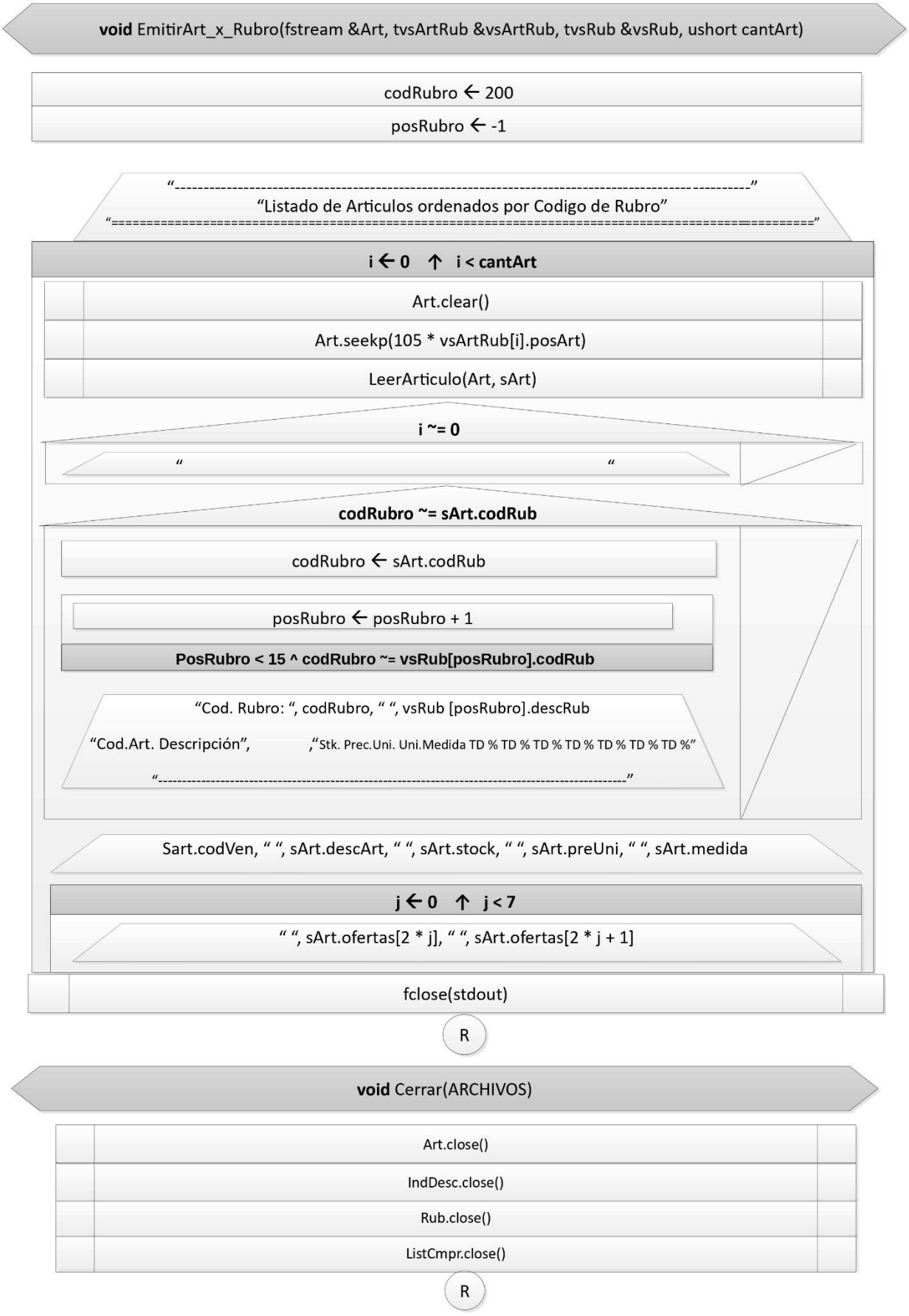

string Replicate(char E:car, ushort E:n)
resultado ← ""
i ← 0 ↑ i < n
resultado ← resultado + car
retornar resultado
R


void Abrir(ARCHIVOS)
Art.open("Articulos.txt")
IndDesc.open("IndDescripArt.txt")
Rub.open("Rubros.txt")
ListCmpr.open("ListaCompras.txt")
R


void VolcarArchivos(ARCHIVOS, REGISTROS, ushort S:cantArt, ushort S:cantCmpr)
cantArt ← 0
cantCmpr ← 0
LeerArticulo(Art, sArt) ^ cantArt < MAX_ART
vsArtRub[cantArt].codRub ← sArt.codRub
vsArtRub[cantArt].posArt ← cantArt
cantArt ← cantArt + 1
i ← 0 ↑ LeerDescripcion(IndDesc, sIndDesc) ^ i < cantArt
vsIndDesc[i] ← sIndDesc
i ← 0 ↑ LeerRubro(Rub, sRub) ^ i < CANT_RUB
vsRub[i] ← sRub
LeerCompra(ListCmpr, sCompra) ^ cantCmpr < MAX_COMPRAS
vsListCmpr[cantCmpr] ← sCompra
cantCmpr ← cantCmpr + 1
OrdxBur(vsArtRub, cantArt)
R


void ProcCompras(fstream &Art, tvsIndDesc &vsIndDesc, tvsListCmpr &vsListCmpr, ushort cantArt, ushort cantCmpr)
i ← 0 ↑ i < cantCmpr
strcpy(descBuscada, vsListCmpr [i].descArt)
pos ← BusBinVec(vsIndDesc, descBuscada, cantArt - 1)
pos ~= -1 ^ vsIndDesc[pos].estado
posArt ← vsIndDesc[pos].posArt
Art.clear()
Art.seekp(105 * posArt)
LeerArticulo(Art, sArt)
sArt.stock >= vsListCmpr[i].cantReq
sArt.stock ←sArt.stock - vsListCmpr[i].cantReq
vsListCmpr[i].cantReq ← sArt.stock
sArt.stock ← 0
vsListCmpr[i].cantReq ← 0
Art.clear()
Art.seekp(105 * posArt)
ActLinea(Art, vsArt[posArt])
R


void EmitirTicket(fstream &Art, tvsIndDesc &vsIndDesc, tvsListCmpr &vsListCmpr, ushort cantArt, ushort cantCmpr)
impTot ← 0.0f
impTotDesto ← 0.0f
freopen("ticket.txt", "w", stdout)
CabeceraTicket(ds)
i ← 0 ↑ i < cantCmpr
vsListCmpr[i].cantReq > 0
pos ← BusBinVec(vsIndDesc, vsListCmpr[i].descArt, cantArt - 1);
pos > -1
Art.clear()
Art. seekp(105 * vsIndDesc[pos].posArt);
LeerArticulo(Art, sArt);
cant ← vsListCmpr[i].cantReq
precio ← sArt.preUni
subtotal ← cant * precio
tipo ← sArt.ofertas[(ds - 1) * 2]
porc ← sArt.ofertas[(ds - 1) * 2 + 1]
descuento ← 0.0f
tipo >= 2 ^ tipo <= 7
descuento ← subtotal * porc / 100.0f
tipo
case 2
strcpy(strDesc, "Promo")
case 3
strcpy(strDesc, "Marca")
case 4
strcpy(strDesc, "Jub.")
case 5
strcpy(strDesc, "Comunid.")
case 6
strcpy(strDesc, "MercPago")
case 7
strcpy(strDesc, "ANSES")
default
strcpy(strDesc, "SinPromo")
" x $ ", precio
sArt.descArt, " ", sArt.medida
Sart.codVen, "$ ", subtotal
descuento > 0.0f
StrDesc, porc, "$ ", -descuento
impTot ← impTot + subtotal
impTotDesto ← impTotDesto + descuento
impTotConDesto ← impTot - impTotDesto
PieTicket(impTot, impTotDesto, impTotConDesto)
R


void EmitirArt_x_Rubro(fstream &Art, tvsArtRub &vsArtRub, tvsRub &vsRub, ushort cantArt)
codRubro ← 200
posRubro ← -1
"----------"
"Listado de Articulos ordenados por Codigo de Rubro"
"=========="
i ← 0 ↑ i < cantArt
Art.clear()
Art.seekp(105 * vsArtRub[i].posArt)
LeerArticulo(Art, sArt)
i ~= 0
" "
codRubro ~= sArt.codRub
codRubro ← sArt.codRub
posRubro ← posRubro + 1
PosRubro < 15 ^ codRubro ~= vsRub[posRubro].codRub
"Cod. Rubro: ", codRubro, " ", vsRub [posRubro].descRub
"Cod.Art. Descripción", ,"Stk. Prec.Uni. Uni.Medida TD % TD % TD % TD % TD % TD %"
"----------"
Sart.codVen, " ", sArt.descArt, " ", sArt.stock, " ", sArt.preUni, " ", sArt.medida
j ← 0 ↑ j < 7
" ", sArt.ofertas[2 * j], " ", sArt.ofertas[2 * j + 1]
fclose(stdout)
R
void Cerrar(ARCHIVOS)
Art.close()
IndDesc.close()
Rub.close()
ListCmpr.close()
R

Código

```
001: /*
002:  - Nombre  del  programa: TP1V2_K1021G10_HOLM FEDERICO.cpp
003:  - fecha  entrega: 03/09/2025
004:  - Nro. versión: 2
005:  - Breve comentario del objetivo del programa:
006:    Este programa gestiona un sistema de ventas de artículos, permitiendo
007:    registrar artículos, rubros, descripciones y compras. Genera tickets
008:    de compra y listados de artículos por rubro, aplicando descuentos según
009:    promociones. Utiliza archivos para almacenar y recuperar datos.
010:
011:
012:  - Curso: Algoritmos y Estructuras de Datos
013:  - Comision: K1021
014:  - Turno: Mañana
015:  - Docente: Lic. Hugo A. Cuello
016:
017:  Integrantes (Apellido, Nombre):
018:     Almada, Tomas
019:     Baquero, Francisco
020:     Barcala Roca, Santiago
021:     Cejas, Facundo Javier
022:     Dominguez, Joaquin Ezequiel
023:     Holm, Federico
024:     Incutti, Mateo
025:     Mampaso Romero, Brayan
026:     Soria, Francisco
027:
028:  - Nombre del compilador: Borland C++ V.5.5
029:  */
030: #include <cmath>
031: #include <cstdio>
032: #include <cstring>
033: #include <ctime>
034: #include <fstream>
035: #include <iomanip>
036: #include <iostream>
037: #include <string>
038:
039: using namespace std;
040:
041: typedef char str30[31];
042: typedef char str20[21];
043: typedef char str10[11];
044: typedef unsigned short ushort;
045:
046: struct tsArt {
047:   int codVen;
048:   short codRub;
049:   str30 descArt;
050:   ushort stock;
051:   float preUni;
052:   str10 medida;
053:   short ofertas[14];
054: };
055:
056: struct tsIndDesc {
057:   str30 descArt;
058:   int posArt;
059:   bool estado;
060: };
061:
062: struct tsRub {
063:   short codRub;
064:   str20 descRub;
065: };
066:
067: struct tsCompra {
068:   str30 descArt;
```

```
069:   short cantReq;
070: };
071:
072: struct tsArtRub {
073:   short codRub;
074:   int posArt;
075: };
076:
077: const ushort MAX_ART = 10000;
078: const ushort CANT_RUB = 15;
079: const ushort MAX_COMPRAS = 100;
080: typedef tsArtRub tvsArtRub[MAX_ART];
081: typedef tsIndDesc tvsIndDesc[MAX_ART];
082: typedef tsRub tvsRub[CANT_RUB];
083: typedef tsCompra tvsListCmpr[MAX_COMPRAS];
084:
085: #define ARCHIVOS \
086:   fstream &Art, ifstream &IndDesc, ifstream &Rub, ifstream &ListCmpr
087: #define REGISTROS                                              \
088:   tvsArtRub &vsArtRub, tvsIndDesc &vsIndDesc, tvsRub &vsRub, \
089:       tvsListCmpr &vsListCmpr
090:
091: long GetTime(int &hh, int &mm, int &ss);
092: long GetDate(int &year, int &mes, int &dia, int &ds);
093: bool LeerArticulo(fstream &Art, tsArt &sArt);
094: bool LeerIndDescrip(ifstream &IndDesc, tsIndDesc &sIndDesc);
095: bool LeerRubro(ifstream &Rub, tsRub &sRub);
096: bool LeerCompra(ifstream &ListCmpr, tsCompra &sCompra);
097: void PieTicket(float impTot, float impTotDesto, float impTotConDesto);
098: void CabeceraTicket(int &ds);
099: void OrdxBur(tvsArtRub &vsArt, ushort card);
100: void IntCmb(tsArtRub &sElem1, tsArtRub &sElem2);
101: void ActLinea(fstream &Art, tsArt &sArt);
102: int BusBinVec(tvsIndDesc &vsIndDesc, str30 descArt, ushort ult);
103: string Replicate(char car, ushort n);
104: void Abrir(ARCHIVOS);
105: void VolcarArchivos(ARCHIVOS, REGISTROS, ushort &cantArt, ushort &cantCmpr);
106: void ProcCompras(fstream &Art, tvsIndDesc &vsIndDesc, tvsListCmpr &vsListCmpr,
107:                  ushort cantArt, ushort cantCmpr);
108: void EmitirTicket(fstream &Art, tvsIndDesc &vsIndDesc, tvsListCmpr &vsListCmpr,
109:                   ushort cantArt, ushort cantCmpr);
110: void EmitirArt_x_Rubro(fstream &Art, tvsArtRub &vsArtRub, tvsRub &vsRub,
111:                        ushort cantArt);
112: void Cerrar(ARCHIVOS);
113:
114: int main() {
115:   tvsArtRub vsArtRub;
116:   tvsIndDesc vsIndDesc;
117:   tvsRub vsRub;
118:   tvsListCmpr vsListCmpr;
119:   fstream Art;
120:   ifstream IndDesc, Rub, ListCmpr;
121:   ushort cantArt, cantCmpr;
122:
123:   Abrir(Art, IndDesc, Rub, ListCmpr);
124:   VolcarArchivos(Art, IndDesc, Rub, ListCmpr, vsArtRub, vsIndDesc, vsRub,
125:                  vsListCmpr, cantArt, cantCmpr);
126:   ProcCompras(Art, vsIndDesc, vsListCmpr, cantArt, cantCmpr);
127:   EmitirTicket(Art, vsIndDesc, vsListCmpr, cantArt, cantCmpr);
128:   EmitirArt_x_Rubro(Art, vsArtRub, vsRub, cantArt);
129:   Cerrar(Art, IndDesc, Rub, ListCmpr);
130:   return 0;
131: }
132:
133: long GetTime(int &hh, int &mm, int &ss) {
134:   time_t rawtime;
135:   struct tm *timeinfo;
136:
137:   time(&rawtime);
138:   timeinfo = localtime(&rawtime);
139:   hh = timeinfo->tm_hour;
140:   mm = timeinfo->tm_min;
```

```
141:   ss = timeinfo->tm_sec;
142:   return timeinfo->tm_hour * 10000 + timeinfo->tm_min * 100 + timeinfo->tm_sec;
143: }  // GetTime
144:
145: long GetDate(int &year, int &mes, int &dia, int &ds) {
146:   time_t rawtime;
147:   struct tm *timeinfo;
148:
149:   time(&rawtime);
150:   timeinfo = localtime(&rawtime);
151:   year = 1900 + timeinfo->tm_year;
152:   mes = 1 + timeinfo->tm_mon;
153:   dia = timeinfo->tm_mday;
154:   ds = 1 + timeinfo->tm_wday;
155:   return (1900 + timeinfo->tm_year) * 10000 + (1 + timeinfo->tm_mon) * 100 +
156:          timeinfo->tm_mday;
157: }  // GetDate
158:
159: bool LeerArticulo(fstream &Art, tsArt &sArt) {
160:   Art >> sArt.codVen >> sArt.codRub;
161:   Art.ignore();
162:   Art.get(sArt.descArt, 31);
163:   Art >> sArt.stock >> sArt.preUni;
164:   Art.ignore();
165:   Art.get(sArt.medida, 11);
166:   for (short i = 0; i < 14; i++)
167:     Art >> sArt.ofertas[i];
168:   Art.ignore(2, '\n');
169:   return Art.good();
170: }  // LeerArticulo
171:
172: bool LeerIndDescrip(ifstream &IndDesc, tsIndDesc &sIndDesc) {
173:   IndDesc.get(sIndDesc.descArt, 31);
174:   IndDesc >> sIndDesc.posArt >> sIndDesc.estado;
175:   IndDesc.ignore(2, '\n');
176:   strlwr(sIndDesc.descArt);
177:   return IndDesc.good();
178: }  // LeerIndDescrip
179:
180: bool LeerRubro(ifstream &Rub, tsRub &sRub) {
181:   Rub >> sRub.codRub;
182:   Rub.ignore();
183:   Rub.get(sRub.descRub, 21);
184:   Rub.ignore(2, '\n');
185:   return Rub.good();
186: }  // LeerRubro
187:
188: bool LeerCompra(ifstream &ListCmpr, tsCompra &sCompra) {
189:   ListCmpr.get(sCompra.descArt, 31);
190:   ListCmpr >> sCompra.cantReq;
191:   ListCmpr.ignore(2, '\n');
192:   return ListCmpr.good();
193: }  // LeerCompra
194:
195: void PieTicket(float impTot, float impTotDesto, float impTotConDesto) {
196:   float pagoUsuario = impTotConDesto;  // El comprador paga exacto
197:
198:   float vuelto = pagoUsuario - impTotConDesto;
199:
200:   cout << setw(42) << left << "SubTot. sin descuentos....:" << "$ " << setw(10)
201:        << right << impTot << '\n'
202:        << setw(42) << left << "Descuentos por promociones:" << "$ " << setw(10)
203:        << right << -impTotDesto << '\n'
204:        << Replicate('=', 54) << '\n'
205:        << setw(42) << left << "T O T A L" << "$ " << setw(10) << right
206:        << impTotConDesto << '\n'
207:        << Replicate('=', 54) << '\n'
208:        << setw(42) << left << "Su pago con Mercado Pago:" << "$ " << setw(10)
209:        << right << pagoUsuario << endl
210:        << setw(42) << left << "Su vuelto:" << "$ " << setw(10) << right
211:        << vuelto << '\n'
212:        << Replicate(' ', 8) << "G R A C I A S  P O R  S U  C O M P R A\n"
```

```
213:        << Replicate(' ', 8) << "Para consultas, sugerencias o reclamos\n"
214:        << Replicate(' ', 8) << "comunicarse al correo infoKotto.com.ar";
215: }  // PieTicket
216:
217: void CabeceraTicket(int &ds) {
218:   int hh, mm, ss, anio, mes, dia;
219:   GetTime(hh, mm, ss);
220:   GetDate(anio, mes, dia, ds);
221:
222:   const char *diasSemana[] = {"Domingo", "Lunes",   "Martes", "Miercoles",
223:                               "Jueves",  "Viernes", "Sabado"};
224:
225:   cout << "K O T T O\n"
226:        << "Yo te reconozco\n"
227:        << "SUC 170\n"
228:        << "XXXXXX...X 9999\n"
229:        << "XX...X\n"
230:        << "C.U.I.T. 99-99999999-9\n"
231:        << setfill('0') << "Fecha: " << diasSemana[ds - 1] << " " << setw(2)
232:        << right << dia << "/" << setw(2) << mes << "/" << setw(4) << anio
233:        << "\nHora: " << setw(2) << hh << ":" << setw(2) << mm << ":" << setw(2)
234:        << ss << "\nNro. Ticket: 9999-99999999\n"
235:        << "Nro. Caja: 9999\n"
236:        << Replicate('-', 54) << "\nF A C T U R A - B\n"
237:        << "ORIGINAL\n"
238:        << Replicate('-', 54) << endl;
239:
240: }  // CabeceraTicket
241:
242: void OrdxBur(tvsArtRub &vsArtRub, ushort card) {
243:   bool hayCambios;
244:   ushort k = 0;
245:
246:   do {
247:     hayCambios = false;
248:     k++;
249:
250:     for (ushort i = 0; i < card - k; i++) {
251:       if (vsArtRub[i].codRub > vsArtRub[i + 1].codRub) {
252:         IntCmb(vsArtRub[i], vsArtRub[i + 1]);
253:         hayCambios = true;
254:       }
255:     }
256:   } while (hayCambios);
257: }  // OrdxBur
258:
259: void IntCmb(tsArtRub &sElem1, tsArtRub &sElem2) {
260:   tsArtRub auxiliar = sElem1;
261:   sElem1 = sElem2;
262:   sElem2 = auxiliar;
263: }  // IntCmb
264:
265: void ActLinea(fstream &Art, tsArt &sArt) {
266:   Art << setw(8) << sArt.codVen << ' ' << setw(2) << sArt.codRub << ' '
267:       << setw(30) << sArt.descArt << ' ' << setw(4) << sArt.stock << ' '
268:       << setw(9) << sArt.preUni << ' ' << setw(10) << sArt.medida;
269:   for (ushort j = 0; j < 7; j++)
270:     Art << ' ' << sArt.ofertas[2 * j] << ' ' << setw(2)
271:         << sArt.ofertas[2 * j + 1];
272: }  // ActLinea
273:
274: int BusBinVec(tvsIndDesc &vsIndDesc, str30 descArt, ushort ult) {
275:   int li = 0, ls = ult, pm;
276:
277:   strlwr(descArt);
278:
279:   while (li <= ls) {
280:     pm = (li + ls) / 2;
281:
282:     int cmp = strcmp(descArt, vsIndDesc[pm].descArt);
283:
284:     if (cmp == 0) {
```

```
285:       return pm;
286:     } else if (cmp < 0) {
287:       ls = pm - 1;
288:     } else {
289:       li = pm + 1;
290:     }
291:   }
292:
293:   return -1;  // No encontrado
294: }  // BusBinVec
295:
296: string Replicate(char car, ushort n) {
297:   string resultado = "";
298:   for (ushort i = 0; i < n; i++)
299:     resultado += car;
300:   return resultado;
301: }  // Replicate
302:
303: void Abrir(ARCHIVOS) {
304:   Art.open("Articulos.txt");
305:   IndDesc.open("IndDescripArt.txt");
306:   Rub.open("Rubros.txt");
307:   ListCmpr.open("ListaCompras.txt");
308: }  // Abrir
309:
310: void VolcarArchivos(ARCHIVOS, REGISTROS, ushort &cantArt, ushort &cantCmpr) {
311:   tsArt sArt;
312:   tsIndDesc sIndDesc;
313:   tsRub sRub;
314:   tsCompra sCompra;
315:   cantArt = 0;
316:   cantCmpr = 0;
317:
318:   while (LeerArticulo(Art, sArt) && cantArt < MAX_ART) {
319:     vsArtRub[cantArt].codRub = sArt.codRub;
320:     vsArtRub[cantArt].posArt = cantArt;
321:     cantArt++;
322:   }
323:   for (ushort i = 0; LeerIndDescrip(IndDesc, sIndDesc) && i < cantArt; i++)
324:     vsIndDesc[i] = sIndDesc;
325:   for (ushort i = 0; LeerRubro(Rub, sRub) && i < CANT_RUB; i++)
326:     vsRub[i] = sRub;
327:   while (LeerCompra(ListCmpr, sCompra) && cantCmpr < MAX_COMPRAS) {
328:     vsListCmpr[cantCmpr] = sCompra;
329:     cantCmpr++;
330:   }
331:
332:   OrdxBur(vsArtRub, cantArt);
333: }  // VolcarArchivos
334:
335: void ProcCompras(fstream &Art, tvsIndDesc &vsIndDesc, tvsListCmpr &vsListCmpr,
336:                  ushort cantArt, ushort cantCmpr) {
337:   str30 descBuscada;
338:   int pos;
339:   ushort posArt;
340:   tsArt sArt;
341:   Art << fixed << setprecision(2);
342:
343:   for (ushort i = 0; i < cantCmpr; i++) {
344:     strcpy(descBuscada, vsListCmpr[i].descArt);
345:     pos = BusBinVec(vsIndDesc, descBuscada, cantArt - 1);
346:
347:     if (pos != -1 && vsIndDesc[pos].estado) {
348:       posArt = vsIndDesc[pos].posArt;
349:
350:       Art.clear();
351:       Art.seekp(105 * posArt);
352:       LeerArticulo(Art, sArt);
353:
354:       if (sArt.stock >= vsListCmpr[i].cantReq) {
355:         sArt.stock -= vsListCmpr[i].cantReq;
356:       } else {
```

```
357:         vsListCmpr[i].cantReq = sArt.stock;
358:         sArt.stock = 0;
359:       }
360:       Art.clear();
361:       Art.seekp(105 * posArt);
362:       ActLinea(Art, sArt);
363:
364:     } else {
365:       vsListCmpr[i].cantReq = 0;
366:     }
367:   }
368: }  // ProcCompras
369:
370: void EmitirTicket(fstream &Art, tvsIndDesc &vsIndDesc, tvsListCmpr &vsListCmpr,
371:                   ushort cantArt, ushort cantCmpr) {
372:   int ds;
373:   float impTot = 0.0f, impTotDesto = 0.0f;
374:   tsArt sArt;
375:
376:   freopen("Ticket.txt", "w", stdout);
377:   CabeceraTicket(ds);
378:   cout << fixed << setprecision(2) << setfill(' ');
379:
380:   for (ushort i = 0; i < cantCmpr; i++) {
381:     if (vsListCmpr[i].cantReq > 0) {
382:       int pos = BusBinVec(vsIndDesc, vsListCmpr[i].descArt, cantArt - 1);
383:       if (pos > -1) {
384:         Art.clear();
385:         Art.seekp(105 * vsIndDesc[pos].posArt);
386:         LeerArticulo(Art, sArt);
387:
388:         ushort cant = vsListCmpr[i].cantReq;
389:         float precio = sArt.preUni;
390:         float subtotal = cant * precio;
391:
392:         ushort tipo = sArt.ofertas[(ds - 1) * 2];
393:         ushort porc = sArt.ofertas[(ds - 1) * 2 + 1];
394:         float descuento = 0.0f;
395:         str10 strDesc;
396:
397:         if (tipo >= 2 && tipo <= 7)  // Solo aplicar si es válido
398:           descuento = subtotal * porc / 100.0f;
399:
400:         switch (tipo) {
401:           case 2:
402:             strcpy(strDesc, "Promo");
403:             break;
404:           case 3:
405:             strcpy(strDesc, "Marca");
406:             break;
407:           case 4:
408:             strcpy(strDesc, "Jub.");
409:             break;
410:           case 5:
411:             strcpy(strDesc, "Comunid.");
412:             break;
413:           case 6:
414:             strcpy(strDesc, "MercPago");
415:             break;
416:           case 7:
417:             strcpy(strDesc, "ANSES");
418:             break;
419:           default: // case 0:
420:             strcpy(strDesc, "SinPromo");
421:             break;
422:         }
423:
424:         cout << setw(2) << right << cant << " x $ " << setw(9) << precio << '\n'
425:              << setw(30) << left << sArt.descArt << ' ' << setw(10)
426:              << sArt.medida << '\n'
427:              << setw(8) << right << sArt.codVen << setw(36) << "$ " << setw(10)
428:              << subtotal << '\n';
```

```
429:
430:         if (descuento > 0.0f) {
431:           cout << setw(12) << left << strDesc << setw(5) << right << porc
432:                << setw(27) << "$ " << setw(10) << -descuento << '\n';
433:         }
434:
435:         impTot += subtotal;
436:         impTotDesto += descuento;
437:         cout << '\n';
438:       }
439:     }
440:   }
441:
442:   float impTotConDesto = impTot - impTotDesto;
443:
444:   PieTicket(impTot, impTotDesto, impTotConDesto);
445: }
446:
447: void EmitirArt_x_Rubro(fstream &Art, tvsArtRub &vsArtRub, tvsRub &vsRub,
448:                        ushort cantArt) {
449:   cout << setfill(' ') << setprecision(2) << fixed;
450:   ushort codRubro = 200;
451:   short posRubro = -1;
452:   tsArt sArt;
453:
454:   cout << Replicate('\n', 10) << '\n'  // Separa el Ticket del Listado
455:        << Replicate('-', 100) << '\n'
456:        << Replicate(' ', floor((100.0 - 50.0) / 2.0))
457:        << "Listado de Articulos ordenados por Codigo de Rubro"
458:        << Replicate(' ', ceil((100.0 - 50.0) / 2.0)) << '\n'
459:        << Replicate('=', 100) << '\n';
460:   for (ushort i = 0; i < cantArt; i++) {
461:     Art.clear();
462:     Art.seekp(105 * vsArtRub[i].posArt);
463:     LeerArticulo(Art, sArt);
464:     if (i != 0)
465:       cout << '\n';
466:     if (codRubro != sArt.codRub) {
467:       codRubro = sArt.codRub;
468:       do {
469:         posRubro++;
470:       } while (posRubro < 15 && codRubro != vsRub[posRubro].codRub);
471:
472:       cout << "\nCod. Rubro: " << codRubro << ' ' << vsRub[posRubro].descRub
473:            << "\nCod.Art. Descripcion" << Replicate(' ', 20)
474:            << "Stk. Prec.Uni. Uni.Medida TD % TD % TD % TD % TD % TD % TD %\n"
475:            << Replicate('-', 100) << '\n';
476:     }
477:     cout << setw(8) << right << sArt.codVen << ' ' << setw(30) << left
478:          << sArt.descArt << ' ' << setw(4) << right << sArt.stock << ' '
479:          << setw(9) << right << sArt.preUni << ' ' << setw(10) << left
480:          << sArt.medida;
481:     for (ushort j = 0; j < 7; j++)
482:       cout << ' ' << sArt.ofertas[2 * j] << ' ' << setw(2) << right
483:            << sArt.ofertas[2 * j + 1];
484:   }
485:   fclose(stdout);
486: }  // EmitirArt_x_Rubro
487:
488: void Cerrar(ARCHIVOS) {
489:   Art.close();
490:   IndDesc.close();
491:   Rub.close();
492:   ListCmpr.close();
493: }  // Cerrar
```

Archivos de entrada

Articulos.txt:

```
98024047  1 Banana Cavendish                0   1193.41 kg         7 58 7 13 7 13 4  8 6 80 5 98 3 15
79832716 12 Salsa de Tomate             1893   6937.68 unidad     5 57 5 52 6  2 6 73 1  5 6 82 1 44
44255999 12 Mostaza                     6195    399.88 unidad     5 76 3 33 5 72 2 39 3 47 2 72 6 81
77811558 10 Gomitas Frutales            9439   3692.72 unidad     7 84 1 96 6 91 7 22 7 87 2 86 1 37
71843253 14 Aceite de Coco              7123   1055.04 l          5 72 4 51 4 80 2 64 1 41 2 10 4 30
79944212  5 Pepsi Max                   4347   3035.10 l          2 66 7 61 3 10 3 93 4 29 4 81 5  7
15742093 15 Atun en Aceite              4262   6370.83 unidad     4 29 4 22 3 20 2  8 2 36 1 25 6 96
75287095  3 Queso Mozzarella            2820   5897.99 l          6 15 4 43 3 77 2 59 2 53 7 39 3 47
96060718  6 Atun Fresco                 3727   4384.19 kg         2 65 6 33 5 95 7 69 2 25 4 54 6 44
19338712 11 Corn Flakes                 7535   4249.51 kg         2 30 1 52 3 36 4  2 6 80 7 39 3 58
10840510  8 Papel Higienico             5109   3415.63 unidad     2 56 3  2 3 79 5 82 7  4 4 55 1 81
99496819  4 Pechuga de Pollo            9225   5718.59 kg         3 33 3 56 3 63 4  5 4 34 5 87 3 12
90880352  2 Tomate Perita               1579   1575.23 kg         3 44 2 78 7  0 6 12 7 62 3 88 3 38
65758412  6 Salmon Fresco               8803   8609.23 kg         7 69 7  7 4 20 2 80 5 10 3 12 2  6
28853710  8 Shampoo                     8496   6525.76 unidad     2 30 1 71 5 75 4 58 6 10 3 21 7 70
68466733  7 Baguette Francesa           4985   5911.89 unidad     2  9 2 89 6 18 1 17 5 58 3 18 3 61
29319846 10 Chocolate con Leche         4569   8125.42 unidad     7 54 4 65 7 45 2 28 3 72 3 61 1 34
30540322  4 Costilla de Cerdo           1940   2244.21 kg         6 41 6 91 7  1 2 88 6 32 7 15 1 68
19078262 15 Arvejas                     2343   7332.64 unidad     4 70 1 54 4 40 4 46 4 64 4 79 5 89
81807170  4 Filet de Res                1814   8042.99 kg         5 45 2 57 6  4 3  4 2 11 2 52 2 11
 3772978  3 Leche Entera                8566   6664.69 l          1  0 2 87 7  8 6 20 5 89 6 99 3 99
99840726 10 Caramelos Surtidos            47   9205.07 unidad     6 55 6 75 4 63 4 30 7 45 1 56 4 98
23880006  2 Lechuga Romana               326   3341.80 kg         6  7 1 96 7 19 6 80 3 25 2 52 5 26
83271845 14 Aceite de Oliva             6614   3005.36 l          7 75 1 82 7 92 3 82 5 44 4 78 3 13
18565176  9 Suavizante                  6737   2687.28 unidad     1 97 3 30 7 36 3 74 5 77 4 83 1 31
57622019 14 Aceite de Girasol           4792   6645.97 l          3 64 5 73 5 34 7 88 4 18 2 45 1 64
75291495  1 Naranja Valencia            5667   2681.34 kg         2 36 7 23 7  4 7 94 3 67 6 73 1 76
65805349  2 Choclo                      1830   9441.03 kg         4 22 6 53 1 15 4 31 2 77 1 23 1 90
95959493  6 Merluza Congelada           6071   3422.22 kg         3 21 1 20 1 85 5 61 4 13 3 61 2 40
85944160  1 Manzana Red Delicious          5   8922.24 kg         3 49 3  0 2 86 6 74 5 10 2 17 3 45
 2384353 15 Maiz Dulce                  2154   9349.45 unidad     1 82 7 89 3 96 7 31 7 35 6 76 2 34
69233733 11 Granola                     1392   5755.99 kg         4 95 7 53 4 35 6  9 7 26 3 62 7 48
57285423  9 Limpiador Multiusos         9867   1029.55 unidad     1 27 3 17 2 86 7  9 7 80 2 80 3 48
13316861  5 Coca-Cola Zero              9640   7968.79 l          4 56 3 83 2 35 6 73 5 73 4 26 3 48
80786032 13 Helado Vainilla             3499   9071.57 unidad     5 14 1 64 1  7 4 61 3 80 5 34 3 74
37897727 12 Mayonesa                    1989   3664.18 unidad     1 53 1 50 4 34 7 16 7 19 2 53 3 52
82574849  9 Detergente Polvo            2055   6237.69 unidad     2 83 2 61 3 98 5 55 6 84 5 38 5 34
33415557  7 Medialunas                  6934    847.05 unidad     1 32 4 90 3 33 4 20 3 84 1 82 2 96
  766444  8 Jabon Liquido               2911   9198.66 unidad     3 74 3 29 7 48 7 35 4 54 6 85 5 99
24358058 13 Pizza Congelada             8943   3279.19 unidad     3 19 5 70 3 70 2 36 7 49 1 77 6 41
90991857 11 Avena                       6306   5676.29 kg         1 24 3  2 5 80 4 13 2 57 4 32 4 60
49249004  7 Pan Integral                6145   3504.18 unidad     6 93 4 60 5 49 5 90 7 19 7 95 4 25
26304969  5 Agua Mineral                2494   4101.67 l          5 86 6 22 6 60 5 62 1 76 6 49 6 57
43812040  3 Yogur Natural               2367   2761.75 l          2 86 6 90 2 44 7 70 2  2 5 74 5 23
 6643982 13 Verduras Congeladas         4696   5204.42 unidad     5 94 1 36 4 25 6 45 3 83 4  4 6  5
```

IndDescripArt.txt:

```
Aceite de Coco                 4 1
Aceite de Girasol             25 0
Aceite de Oliva               23 1
Agua Mineral                  42 1
Arvejas                       18 1
Atun en Aceite                 6 0
Atun Fresco                    8 0
Avena                         40 0
Baguette Francesa             15 1
Banana Cavendish               0 1
Caramelos Surtidos            21 1
Choclo                        27 0
Chocolate con Leche           16 1
Coca-Cola Zero                33 1
Corn Flakes                    9 1
Costilla de Cerdo             17 1
```

```
Detergente Polvo                36 1
Filet de Res                    19 1
Gomitas Frutales                 3 1
Granola                         31 1
Helado Vainilla                 34 1
Jabon Liquido                   38 1
Leche Entera                    20 0
Lechuga Romana                  22 1
Limpiador Multiusos             32 1
Maiz Dulce                      30 0
Manzana Red Delicious           29 1
Mayonesa                        35 1
Medialunas                      37 1
Merluza Congelada               28 0
Mostaza                          2 1
Naranja Valencia                26 1
Pan Integral                    41 0
Papel Higienico                 10 1
Pechuga de Pollo                11 1
Pepsi Max                        5 1
Pizza Congelada                 39 1
Queso Mozzarella                 7 1
Salmon Fresco                   13 1
Salsa de Tomate                  1 1
Shampoo                         14 1
Suavizante                      24 1
Tomate Perita                   12 0
Verduras Congeladas             44 1
Yogur Natural                   43 1
```

ListaCompras.txt:

```
Banana Cavendish              12
Salsa de Tomate                5
Mostaza                       10
Gomitas Frutales              20
Aceite de Coco                 8
Pepsi Max                     15
Atun en Aceite                 6
Queso Mozzarella               7
Atun Fresco                   11
Corn Flakes                   13
Papel Higienico                9
Pechuga de Pollo               4
Tomate Perita                 14
Salmon Fresco                 10
Shampoo                       12
Baguette Francesa              3
Chocolate con Leche            8
Costilla de Cerdo              5
Arvejas                       17
Filet de Res                  16
Leche Entera                  14
Caramelos Surtidos            19
Lechuga Romana                11
Aceite de Oliva               15
Suavizante                    10
Aceite de Girasol             12
Naranja Valencia               6
Choclo                         9
Merluza Congelada             13
Manzana Red Delicious         10
```

Rubros.txt:

```
1 Frutas
2 Verduras
3 Lacteos
4 Carnes
```

```
 5 Bebidas
 6 Pescados
 7 Panaderia
 8 Higiene
 9 Limpieza
10 Dulces
11 Cereales
12 Salsas
13 Congelados
14 Aceites
15 Enlatados
```

Archivos de Salida

Articulos.txt (Luego de la ejecución del programa):

```
98024047  1 Banana Cavendish                0   1193.41 kg         7 58 7 13 7 13 4  8 6 80 5 98 3 15
79832716 12 Salsa de Tomate              1888   6937.68 unidad     5 57 5 52 6  2 6 73 1  5 6 82 1 44
44255999 12 Mostaza                      6185    399.88 unidad     5 76 3 33 5 72 2 39 3 47 2 72 6 81
77811558 10 Gomitas Frutales             9419   3692.72 unidad     7 84 1 96 6 91 7 22 7 87 2 86 1 37
71843253 14 Aceite de Coco               7115   1055.04 l          5 72 4 51 4 80 2 64 1 41 2 10 4 30
79944212  5 Pepsi Max                    4332   3035.10 l          2 66 7 61 3 10 3 93 4 29 4 81 5  7
15742093 15 Atun en Aceite               4262   6370.83 unidad     4 29 4 22 3 20 2  8 2 36 1 25 6 96
75287095  3 Queso Mozzarella             2813   5897.99 l          6 15 4 43 3 77 2 59 2 53 7 39 3 47
96060718  6 Atun Fresco                  3727   4384.19 kg         2 65 6 33 5 95 7 69 2 25 4 54 6 44
19338712 11 Corn Flakes                  7522   4249.51 kg         2 30 1 52 3 36 4  2 6 80 7 39 3 58
10840510  8 Papel Higienico              5100   3415.63 unidad     2 56 3  2 3 79 5 82 7  4 4 55 1 81
99496819  4 Pechuga de Pollo             9221   5718.59 kg         3 33 3 56 3 63 4  5 4 34 5 87 3 12
90880352  2 Tomate Perita                1579   1575.23 kg         3 44 2 78 7  0 6 12 7 62 3 88 3 38
65758412  6 Salmon Fresco                8793   8609.23 kg         7 69 7  7 4 20 2 80 5 10 3 12 2  6
28853710  8 Shampoo                      8484   6525.76 unidad     2 30 1 71 5 75 4 58 6 10 3 21 7 70
68466733  7 Baguette Francesa            4982   5911.89 unidad     2  9 2 89 6 18 1 17 5 58 3 18 3 61
29319846 10 Chocolate con Leche          4561   8125.42 unidad     7 54 4 65 7 45 2 28 3 72 3 61 1 34
30540322  4 Costilla de Cerdo            1935   2244.21 kg         6 41 6 91 7  1 2 88 6 32 7 15 1 68
19078262 15 Arvejas                      2326   7332.64 unidad     4 70 1 54 4 40 4 46 4 64 4 79 5 89
81807170  4 Filet de Res                 1798   8042.99 kg         5 45 2 57 6  4 3  4 2 11 2 52 2 11
 3772978  3 Leche Entera                 8566   6664.69 l          1  0 2 87 7  8 6 20 5 89 6 99 3 99
99840726 10 Caramelos Surtidos           28   9205.07 unidad     6 55 6 75 4 63 4 30 7 45 1 56 4 98
23880006  2 Lechuga Romana               315   3341.80 kg         6  7 1 96 7 19 6 80 3 25 2 52 5 26
83271845 14 Aceite de Oliva              6599   3005.36 l          7 75 1 82 7 92 3 82 5 44 4 78 3 13
18565176  9 Suavizante                   6727   2687.28 unidad     1 97 3 30 7 36 3 74 5 77 4 83 1 31
57622019 14 Aceite de Girasol            4792   6645.97 l          3 64 5 73 5 34 7 88 4 18 2 45 1 64
75291495  1 Naranja Valencia             5661   2681.34 kg         2 36 7 23 7  4 7 94 3 67 6 73 1 76
65805349  2 Choclo                       1830   9441.03 kg         4 22 6 53 1 15 4 31 2 77 1 23 1 90
95959493  6 Merluza Congelada            6071   3422.22 kg         3 21 1 20 1 85 5 61 4 13 3 61 2 40
85944160  1 Manzana Red Delicious           0   8922.24 kg         3 49 3  0 2 86 6 74 5 10 2 17 3 45
 2384353 15 Maiz Dulce                   2154   9349.45 unidad     1 82 7 89 3 96 7 31 7 35 6 76 2 34
69233733 11 Granola                      1392   5755.99 kg         4 95 7 53 4 35 6  9 7 26 3 62 7 48
57285423  9 Limpiador Multiusos          9867   1029.55 unidad     1 27 3 17 2 86 7  9 7 80 2 80 3 48
13316861  5 Coca-Cola Zero               9640   7968.79 l          4 56 3 83 2 35 6 73 5 73 4 26 3 48
80786032 13 Helado Vainilla              3499   9071.57 unidad     5 14 1 64 1  7 4 61 3 80 5 34 3 74
37897727 12 Mayonesa                     1989   3664.18 unidad     1 53 1 50 4 34 7 16 7 19 2 53 3 52
82574849  9 Detergente Polvo             2055   6237.69 unidad     2 83 2 61 3 98 5 55 6 84 5 38 5 34
33415557  7 Medialunas                   6934    847.05 unidad     1 32 4 90 3 33 4 20 3 84 1 82 2 96
  766444  8 Jabon Liquido                2911   9198.66 unidad     3 74 3 29 7 48 7 35 4 54 6 85 5 99
24358058 13 Pizza Congelada              8943   3279.19 unidad     3 19 5 70 3 70 2 36 7 49 1 77 6 41
90991857 11 Avena                        6306   5676.29 kg         1 24 3  2 5 80 4 13 2 57 4 32 4 60
49249004  7 Pan Integral                 6145   3504.18 unidad     6 93 4 60 5 49 5 90 7 19 7 95 4 25
26304969  5 Agua Mineral                 2494   4101.67 l          5 86 6 22 6 60 5 62 1 76 6 49 6 57
43812040  3 Yogur Natural                2367   2761.75 l          2 86 6 90 2 44 7 70 2  2 5 74 5 23
 6643982 13 Verduras Congeladas          4696   5204.42 unidad     5 94 1 36 4 25 6 45 3 83 4  4 6  5
```

Ticket.txt:

```
K O T T O
Yo te reconozco
SUC 170
XXXXXX...X 9999
XX...X
C.U.I.T. 99-99999999-9
Fecha: Jueves 28/08/2025
Hora: 18:19:32
Nro. Ticket: 9999-99999999
Nro. Caja: 9999
-----------------------------------------------------
F A C T U R A - B
ORIGINAL
-----------------------------------------------------
 5 x $   6937.68
Salsa de Tomate                 unidad
79832716                                $   34688.40

10 x $    399.88
Mostaza                         unidad
44255999                                $    3998.80
Marca          47                       $   -1879.44

20 x $   3692.72
Gomitas Frutales                unidad
77811558                                $   73854.40
ANSES          87                       $  -64253.33

 8 x $   1055.04
Aceite de Coco                  l
71843253                                $    8440.32

15 x $   3035.10
Pepsi Max                       l
79944212                                $   45526.50
Jub.           29                       $  -13202.68

 7 x $   5897.99
Queso Mozzarella                l
75287095                                $   41285.93
Promo          53                       $  -21881.54

13 x $   4249.51
Corn Flakes                     kg
19338712                                $   55243.62
MercPago       80                       $  -44194.90

 9 x $   3415.63
Papel Higienico                 unidad
10840510                                $   30740.67
ANSES           4                       $   -1229.63

 4 x $   5718.59
Pechuga de Pollo                kg
99496819                                $   22874.36
Jub.           34                       $   -7777.28

10 x $   8609.23
Salmon Fresco                   kg
65758412                                $   86092.30
Comunid.       10                       $   -8609.23

12 x $   6525.76
Shampoo                         unidad
28853710                                $   78309.12
MercPago       10                       $   -7830.91

 3 x $   5911.89
Baguette Francesa               unidad
68466733                                $   17735.67
Comunid.       58                       $  -10286.69
```

```
 8 x $   8125.42
Chocolate con Leche            unidad
29319846                                $   65003.36
Marca          72                       $  -46802.42

 5 x $   2244.21
Costilla de Cerdo              kg
30540322                                $   11221.05
MercPago       32                       $   -3590.74

17 x $   7332.64
Arvejas                        unidad
19078262                                $  124654.88
Jub.           64                       $  -79779.12

16 x $   8042.99
Filet de Res                   kg
81807170                                $  128687.84
Promo          11                       $  -14155.66

19 x $   9205.07
Caramelos Surtidos             unidad
99840726                                $  174896.34
ANSES          45                       $  -78703.35

11 x $   3341.80
Lechuga Romana                 kg
23880006                                $   36759.80
Marca          25                       $   -9189.95

15 x $   3005.36
Aceite de Oliva                l
83271845                                $   45080.40
Comunid.       44                       $  -19835.38

10 x $   2687.28
Suavizante                     unidad
18565176                                $   26872.80
Comunid.       77                       $  -20692.06

 6 x $   2681.34
Naranja Valencia               kg
75291495                                $   16088.04
Marca          67                       $  -10778.99

 5 x $   8922.24
Manzana Red Delicious          kg
85944160                                $   44611.20
Comunid.       10                       $   -4461.12

SubTot. sin descuentos....:             $ 1172665.88
Descuentos por promociones:             $ -469134.44
=====================================================
T O T A L                               $  703531.44
=====================================================
Su pago con Mercado Pago:               $  703531.44
Su vuelto:                              $       0.00
        G R A C I A S   P O R   S U   C O M P R A
        Para consultas, sugerencias o reclamos
        comunicarse al correo infoKotto.com.ar
```

```
=============================================================================================================

Cod. Rubro: 1 Frutas
Cod.Art. Descripcion                    Stk. Prec.Uni. Uni.Medida TD % TD % TD % TD % TD % TD % TD %
-------------------------------------------------------------------------------------------------------------
98024047 Banana Cavendish                  0   1193.41 kg         7 58 7 13 7 13 4  8 6 80 5 98 3 15
75291495 Naranja Valencia               5661   2681.34 kg         2 36 7 23 7  4 7 94 3 67 6 73 1 76
85944160 Manzana Red Delicious             0   8922.24 kg         3 49 3  0 2 86 6 74 5 10 2 17 3 45

Cod. Rubro: 2 Verduras
Cod.Art. Descripcion                    Stk. Prec.Uni. Uni.Medida TD % TD % TD % TD % TD % TD % TD %
-------------------------------------------------------------------------------------------------------------
90880352 Tomate Perita                  1579   1575.23 kg         3 44 2 78 7  0 6 12 7 62 3 88 3 38
23880006 Lechuga Romana                  315   3341.80 kg         6  7 1 96 7 19 6 80 3 25 2 52 5 26
65805349 Choclo                         1830   9441.03 kg         4 22 6 53 1 15 4 31 2 77 1 23 1 90

Cod. Rubro: 3 Lacteos
Cod.Art. Descripcion                    Stk. Prec.Uni. Uni.Medida TD % TD % TD % TD % TD % TD % TD %
-------------------------------------------------------------------------------------------------------------
75287095 Queso Mozzarella               2813   5897.99 l          6 15 4 43 3 77 2 59 2 53 7 39 3 47
 3772978 Leche Entera                   8566   6664.69 l          1  0 2 87 7  8 6 20 5 89 6 99 3 99
43812040 Yogur Natural                  2367   2761.75 l          2 86 6 90 2 44 7 70 2  2 5 74 5 23

Cod. Rubro: 4 Carnes
Cod.Art. Descripcion                    Stk. Prec.Uni. Uni.Medida TD % TD % TD % TD % TD % TD % TD %
-------------------------------------------------------------------------------------------------------------
99496819 Pechuga de Pollo               9221   5718.59 kg         3 33 3 56 3 63 4  5 4 34 5 87 3 12
30540322 Costilla de Cerdo              1935   2244.21 kg         6 41 6 91 7  1 2 88 6 32 7 15 1 68
81807170 Filet de Res                   1798   8042.99 kg         5 45 2 57 6  4 3  4 2 11 2 52 2 11

Cod. Rubro: 5 Bebidas
Cod.Art. Descripcion                    Stk. Prec.Uni. Uni.Medida TD % TD % TD % TD % TD % TD % TD %
-------------------------------------------------------------------------------------------------------------
79944212 Pepsi Max                      4332   3035.10 l          2 66 7 61 3 10 3 93 4 29 4 81 5  7
13316861 Coca-Cola Zero                 9640   7968.79 l          4 56 3 83 2 35 6 73 5 73 4 26 3 48
26304969 Agua Mineral                   2494   4101.67 l          5 86 6 22 6 60 5 62 1 76 6 49 6 57

Cod. Rubro: 6 Pescados
Cod.Art. Descripcion                    Stk. Prec.Uni. Uni.Medida TD % TD % TD % TD % TD % TD % TD %
-------------------------------------------------------------------------------------------------------------
96060718 Atun Fresco                    3727   4384.19 kg         2 65 6 33 5 95 7 69 2 25 4 54 6 44
65758412 Salmon Fresco                  8793   8609.23 kg         7 69 7  7 4 20 2 80 5 10 3 12 2  6
95959493 Merluza Congelada              6071   3422.22 kg         3 21 1 20 1 85 5 61 4 13 3 61 2 40

Cod. Rubro: 7 Panaderia
Cod.Art. Descripcion                    Stk. Prec.Uni. Uni.Medida TD % TD % TD % TD % TD % TD % TD %
-------------------------------------------------------------------------------------------------------------
68466733 Baguette Francesa              4982   5911.89 unidad     2  9 2 89 6 18 1 17 5 58 3 18 3 61
33415557 Medialunas                     6934    847.05 unidad     1 32 4 90 3 33 4 20 3 84 1 82 2 96
49249004 Pan Integral                   6145   3504.18 unidad     6 93 4 60 5 49 5 90 7 19 7 95 4 25

Cod. Rubro: 8 Higiene
Cod.Art. Descripcion                    Stk. Prec.Uni. Uni.Medida TD % TD % TD % TD % TD % TD % TD %
-------------------------------------------------------------------------------------------------------------
10840510 Papel Higienico                5100   3415.63 unidad     2 56 3  2 3 79 5 82 7  4 4 55 1 81
28853710 Shampoo                        8484   6525.76 unidad     2 30 1 71 5 75 4 58 6 10 3 21 7 70
  766444 Jabon Liquido                  2911   9198.66 unidad     3 74 3 29 7 48 7 35 4 54 6 85 5 99

Cod. Rubro: 9 Limpieza
Cod.Art. Descripcion                    Stk. Prec.Uni. Uni.Medida TD % TD % TD % TD % TD % TD % TD %
-------------------------------------------------------------------------------------------------------------
18565176 Suavizante                     6727   2687.28 unidad     1 97 3 30 7 36 3 74 5 77 4 83 1 31
57285423 Limpiador Multiusos            9867   1029.55 unidad     1 27 3 17 2 86 7  9 7 80 2 80 3 48
82574849 Detergente Polvo               2055   6237.69 unidad     2 83 2 61 3 98 5 55 6 84 5 38 5 34

Cod. Rubro: 10 Dulces
Cod.Art. Descripcion                    Stk. Prec.Uni. Uni.Medida TD % TD % TD % TD % TD % TD % TD %
-------------------------------------------------------------------------------------------------------------
77811558 Gomitas Frutales               9419   3692.72 unidad     7 84 1 96 6 91 7 22 7 87 2 86 1 37
29319846 Chocolate con Leche            4561   8125.42 unidad     7 54 4 65 7 45 2 28 3 72 3 61 1 34
99840726 Caramelos Surtidos               28   9205.07 unidad     6 55 6 75 4 63 4 30 7 45 1 56 4 98
```

```
Cod. Rubro: 11 Cereales
Cod.Art. Descripcion                    Stk. Prec.Uni. Uni.Medida TD % TD % TD % TD % TD % TD % TD %
-------------------------------------------------------------------------------------------------
19338712 Corn Flakes                    7522   4249.51 kg         2 30 1 52 3 36 4  2 6 80 7 39 3 58
69233733 Granola                        1392   5755.99 kg         4 95 7 53 4 35 6  9 7 26 3 62 7 48
90991857 Avena                          6306   5676.29 kg         1 24 3  2 5 80 4 13 2 57 4 32 4 60

Cod. Rubro: 12 Salsas
Cod.Art. Descripcion                    Stk. Prec.Uni. Uni.Medida TD % TD % TD % TD % TD % TD % TD %
-------------------------------------------------------------------------------------------------
79832716 Salsa de Tomate                1888   6937.68 unidad     5 57 5 52 6  2 6 73 1  5 6 82 1 44
44255999 Mostaza                        6185    399.88 unidad     5 76 3 33 5 72 2 39 3 47 2 72 6 81
37897727 Mayonesa                       1989   3664.18 unidad     1 53 1 50 4 34 7 16 7 19 2 53 3 52

Cod. Rubro: 13 Congelados
Cod.Art. Descripcion                    Stk. Prec.Uni. Uni.Medida TD % TD % TD % TD % TD % TD % TD %
-------------------------------------------------------------------------------------------------
80786032 Helado Vainilla                3499   9071.57 unidad     5 14 1 64 1  7 4 61 3 80 5 34 3 74
24358058 Pizza Congelada                8943   3279.19 unidad     3 19 5 70 3 70 2 36 7 49 1 77 6 41
 6643982 Verduras Congeladas            4696   5204.42 unidad     5 94 1 36 4 25 6 45 3 83 4  4 6  5

Cod. Rubro: 14 Aceites
Cod.Art. Descripcion                    Stk. Prec.Uni. Uni.Medida TD % TD % TD % TD % TD % TD % TD %
-------------------------------------------------------------------------------------------------
71843253 Aceite de Coco                 7115   1055.04 l          5 72 4 51 4 80 2 64 1 41 2 10 4 30
83271845 Aceite de Oliva                6599   3005.36 l          7 75 1 82 7 92 3 82 5 44 4 78 3 13
57622019 Aceite de Girasol              4792   6645.97 l          3 64 5 73 5 34 7 88 4 18 2 45 1 64

Cod. Rubro: 15 Enlatados
Cod.Art. Descripcion                    Stk. Prec.Uni. Uni.Medida TD % TD % TD % TD % TD % TD % TD %
-------------------------------------------------------------------------------------------------
15742093 Atun en Aceite                 4262   6370.83 unidad     4 29 4 22 3 20 2  8 2 36 1 25 6 96
19078262 Arvejas                        2326   7332.64 unidad     4 70 1 54 4 40 4 46 4 64 4 79 5 89
 2384353 Maiz Dulce                     2154   9349.45 unidad     1 82 7 89 3 96 7 31 7 35 6 76 2 34
```