

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
233.220-6
232.984-0
233.215-2
232.921-9
233.029-5
232.930-0
232.992-0
232.941-4

Dominguez ,Joaquin Ezequiel
Holm ,Federico
Incutti ,Mateo
Baquero ,Francisco
Almada ,Tomas
Soria ,Francisco
Barcala Roca ,Santiago
Mampaso Romero ,Brayan
Cejas ,Facundo Javier

jodominguez@frba.utn.edu.ar
fholm@frba.utn.edu.ar
mincutti@frba.utn.edu.ar
fbaquero@frba.utn.edu.ar
toalmada@frba.utn.edu.ar
Frsoria@frba.utn.edu.ar
sbarcalaroca@frba.utn.edu.ar
bmampaso@frba.utn.edu.ar
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 utilizar 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 línea 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

DIAGRAMAS	6
CÓDIGO	17
ARCHIVOS DE ENTRADA.....	24
ARTICULOS.TXT:	24
INDDESCRIPART.TXT:	24
LISTACOMPRAS.TXT:	25
RUBROS.TXT:.....	26
ARCHIVOS DE SALIDA.....	26
LISTADOARTICULOS.TXT:	26
ARTICULOS.TXT (LUEGO DE LA EJECUCIÓN DEL PROGRAMA):.....	27
TICKET.TXT:	28

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 dígit.)	Cod.Rubro (short 2 dígit.)	Descripción Art. (str30)	Stock Actual (ushort, 4 dígit.)
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 dígit.)
------------------	----------------------------------

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

Se pide:

- Volcar y Generar en la memoria **RAM** estructuras estáticas (Tablas):
 - Volcar a una primer tabla todos los datos del archivo **IndDescripArt.Txt**.
 - Volcar a una segunda tabla todos los datos del archivo **ListaCompras.Txt**.
 - 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).
- Procesar la **tabla de Lista de Compras** recorriendo secuencialmente y por cada *descripción* de artículo buscarlo en la **tabla de IndDescripArt**, 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*.
- Emitir el ticket con igual criterio que la **Lista de Compras** de acuerdo al siguiente diseño:

Datos de la Cabecera son:

KOTTO
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

FACTURA - 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	
TOTAL	\$ 999999.99	

Datos del pie son:

Su pago con Tipo Pago:	\$ 999999.99
Su vuelto:	\$ 9.99
GRACIAS POR SU COMPRA	
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 %	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 %	TD %
Cod. Rubro: 5 BEBIDAS CON ALCOHOL												
Cod.Art.	Descripcion	Stk.	Pre.Uni.	U.Med.	TD %	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 %	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 LeerSuf**(*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: TP1V1_K1023G3_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 describe 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 porcentaje 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

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

	<i>tsIndDesc (36B)</i>		
	str30 descArt (31B)	int posArt (4B)	bool estado (1B)
0	"Aceite de Coco"	4	1
1			
2			
1000			
	<i>tsIndDesc tvsIndDesc [MAX_ART] (36000B)</i>		

	<i>tsArt (82B)</i>						
	int codVen (4B)	short codRub (2B)	str30 descArt (31B)	ushort stock (2B)	float preUni (4B)	str10 medida (11B)	short ofertas[14] (28B)
0	15742093	15	"Atun en Aceite"	4262	6370.83	"unidad"	{3,29,3,22,2,20,1,08,1,36,0,25,5,96}
1							
2							
1000							
	<i>tsArt tvsArt [MAX_ART] (82000B)</i>						

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

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

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)	

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 LeerDescripcion(ifstream &IndDesc, tsIndDesc &sIndDesc)

IndDesc.get(sIndDesc.descArt, 31)

IndDesc >> sIndDesc.posArt >> sIndDesc.estado

IndDesc.ignore(2, '\n')

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)

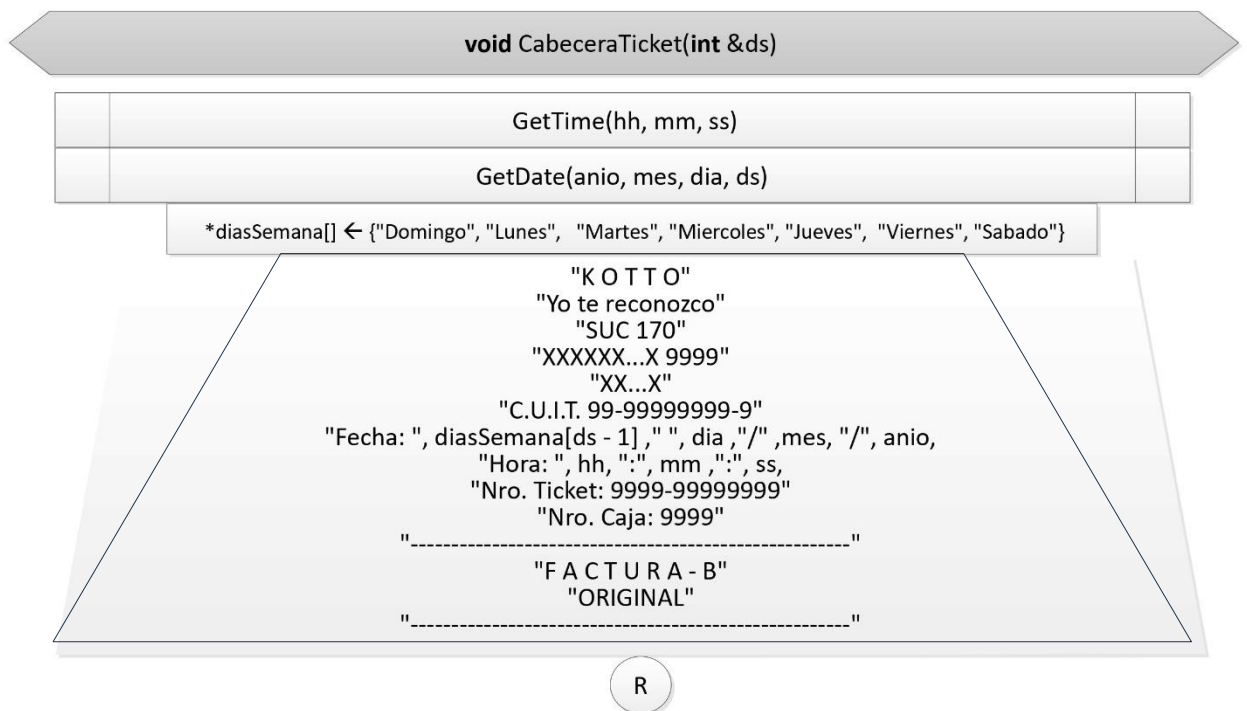
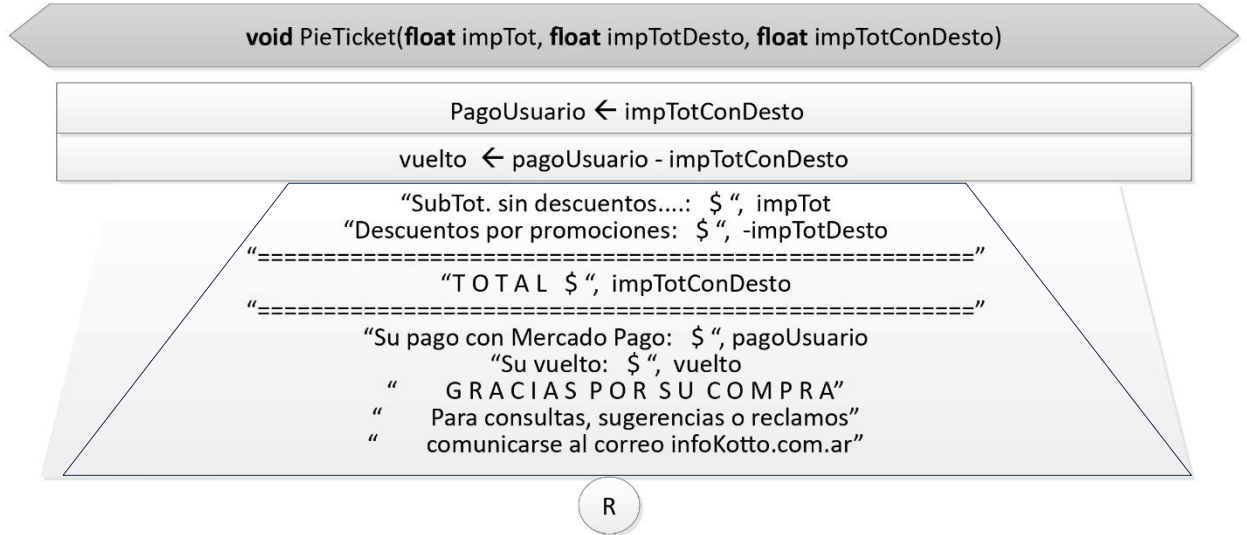
ListCmpr.get(sCompra.descArt, 31)

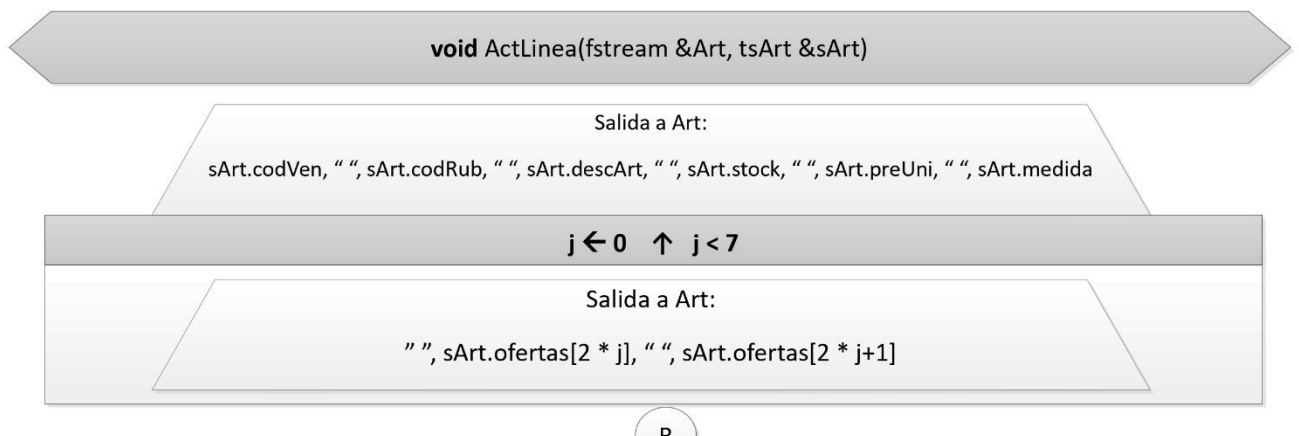
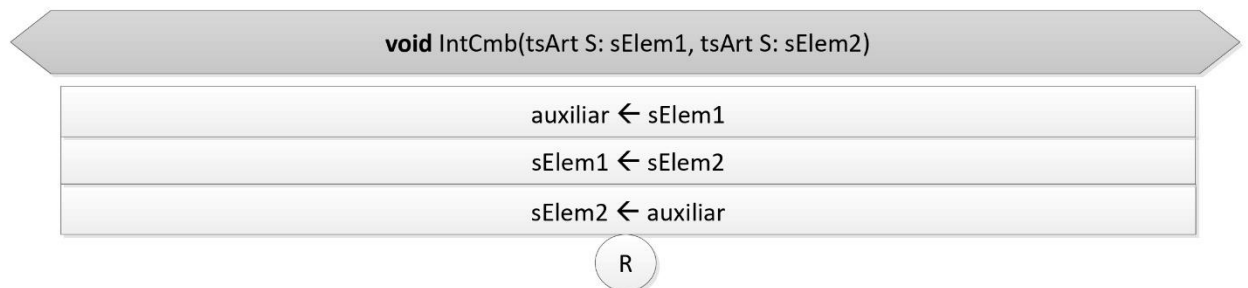
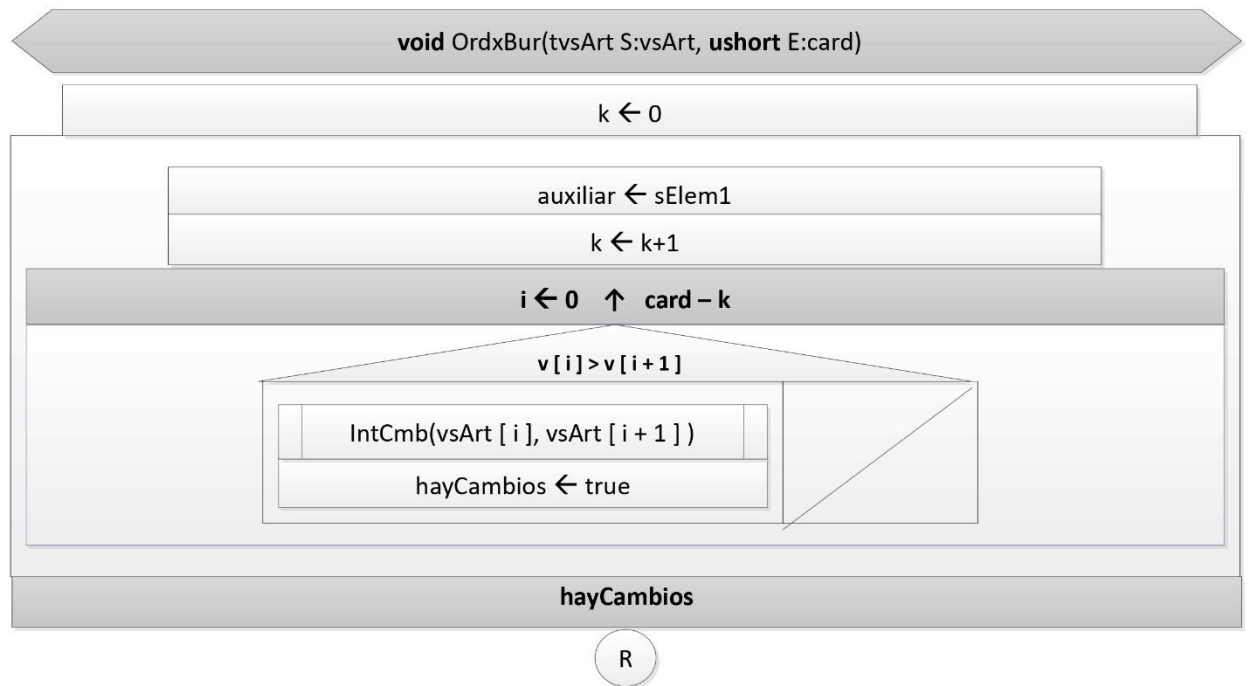
ListCmpr >> sCompra.cantReq

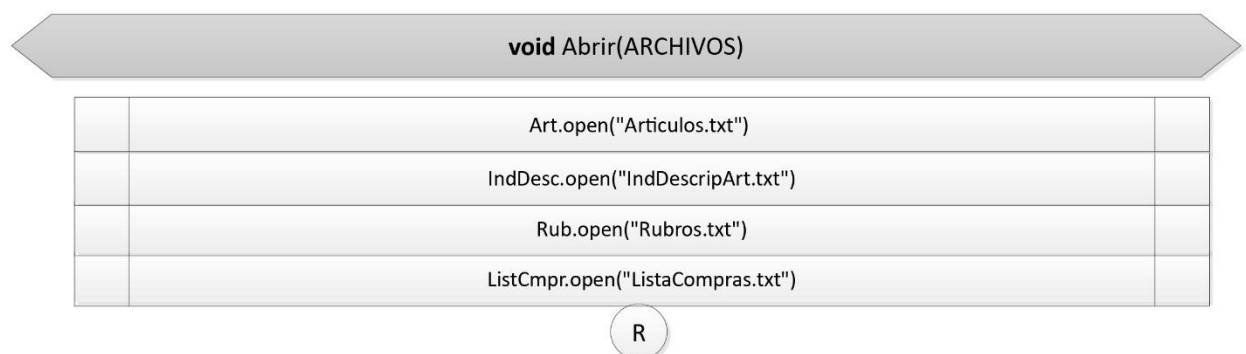
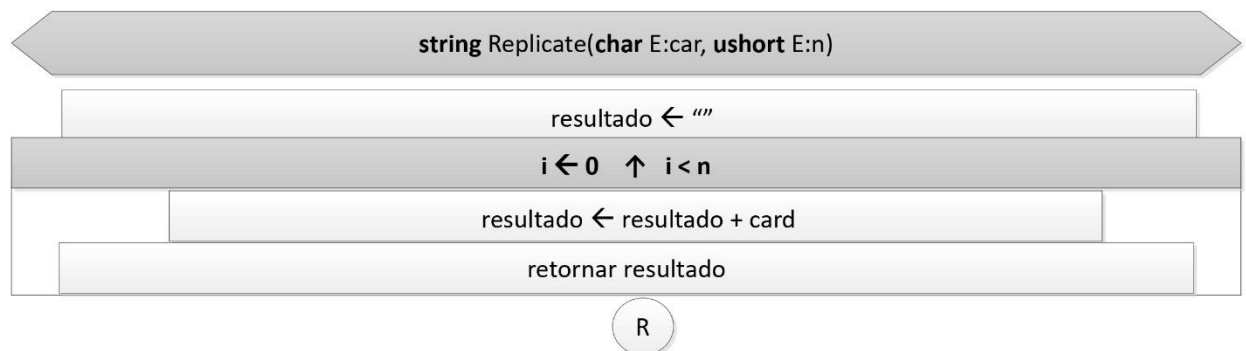
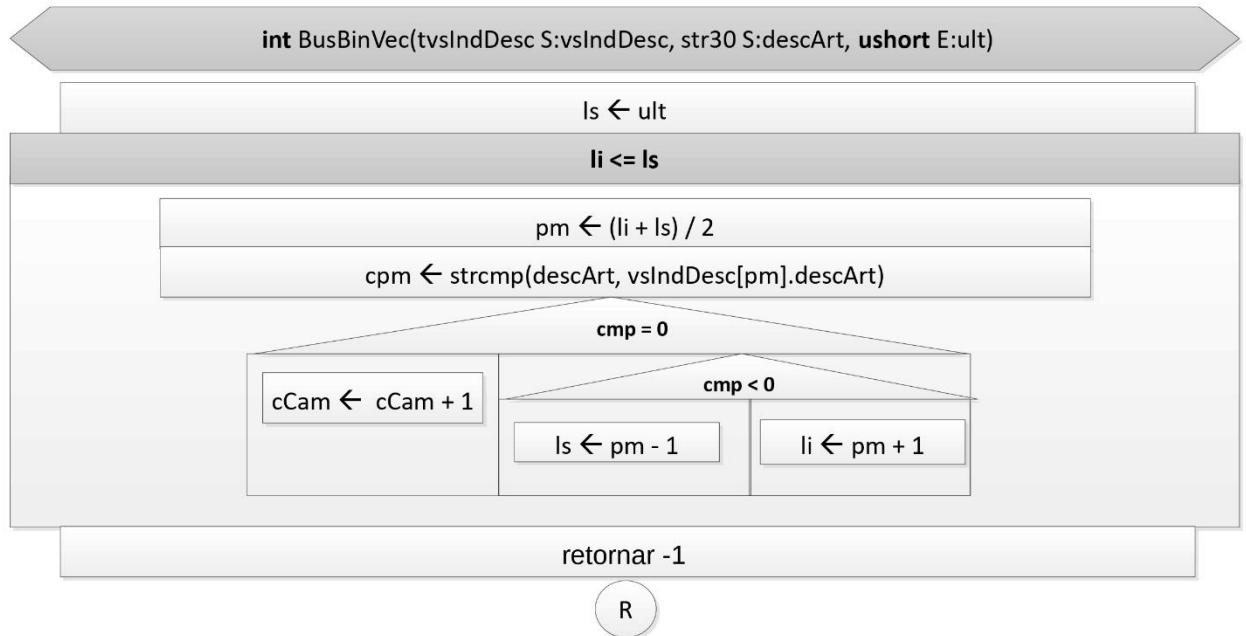
ListCmpr.ignore(2, '\n')

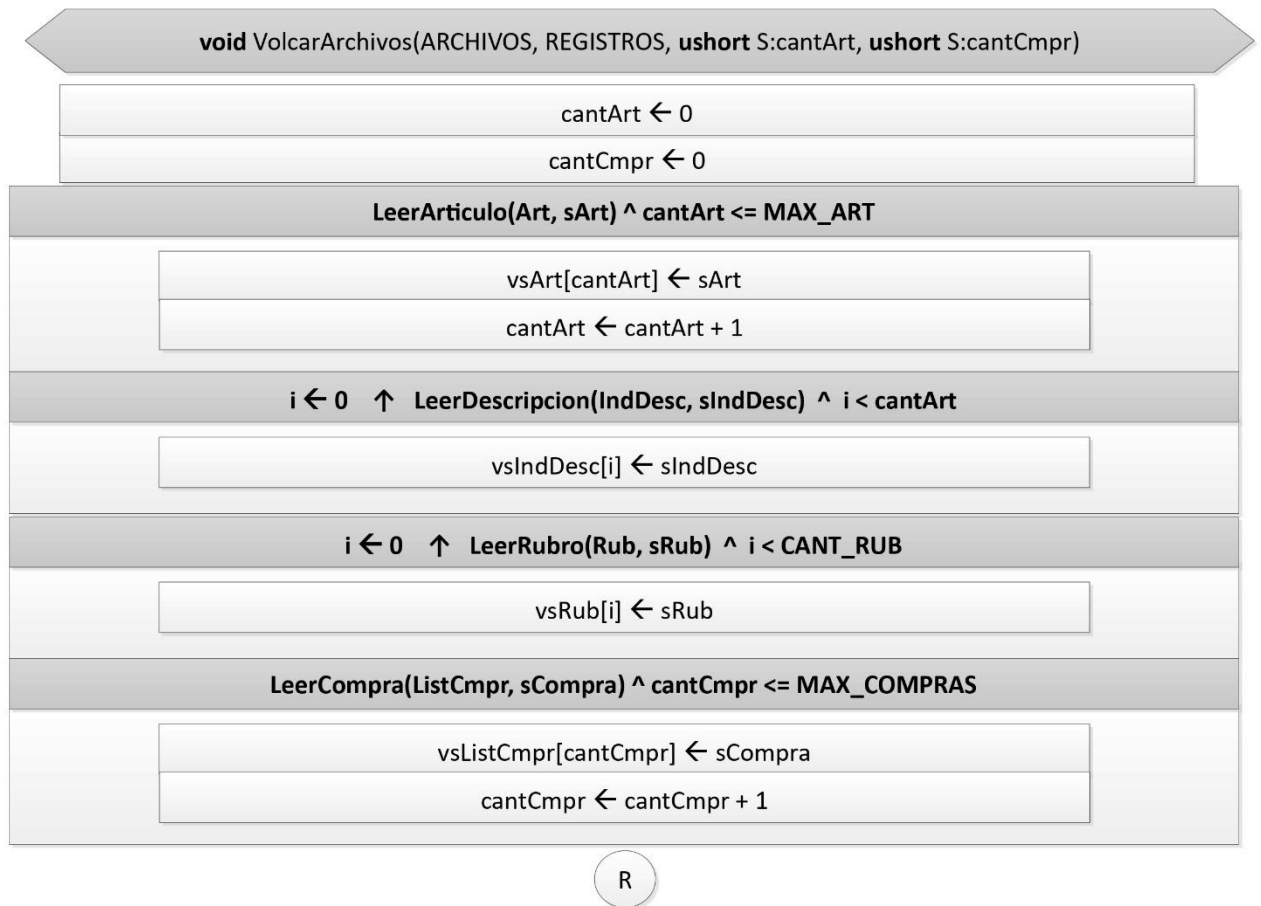
Retornar ListCmpr.good()

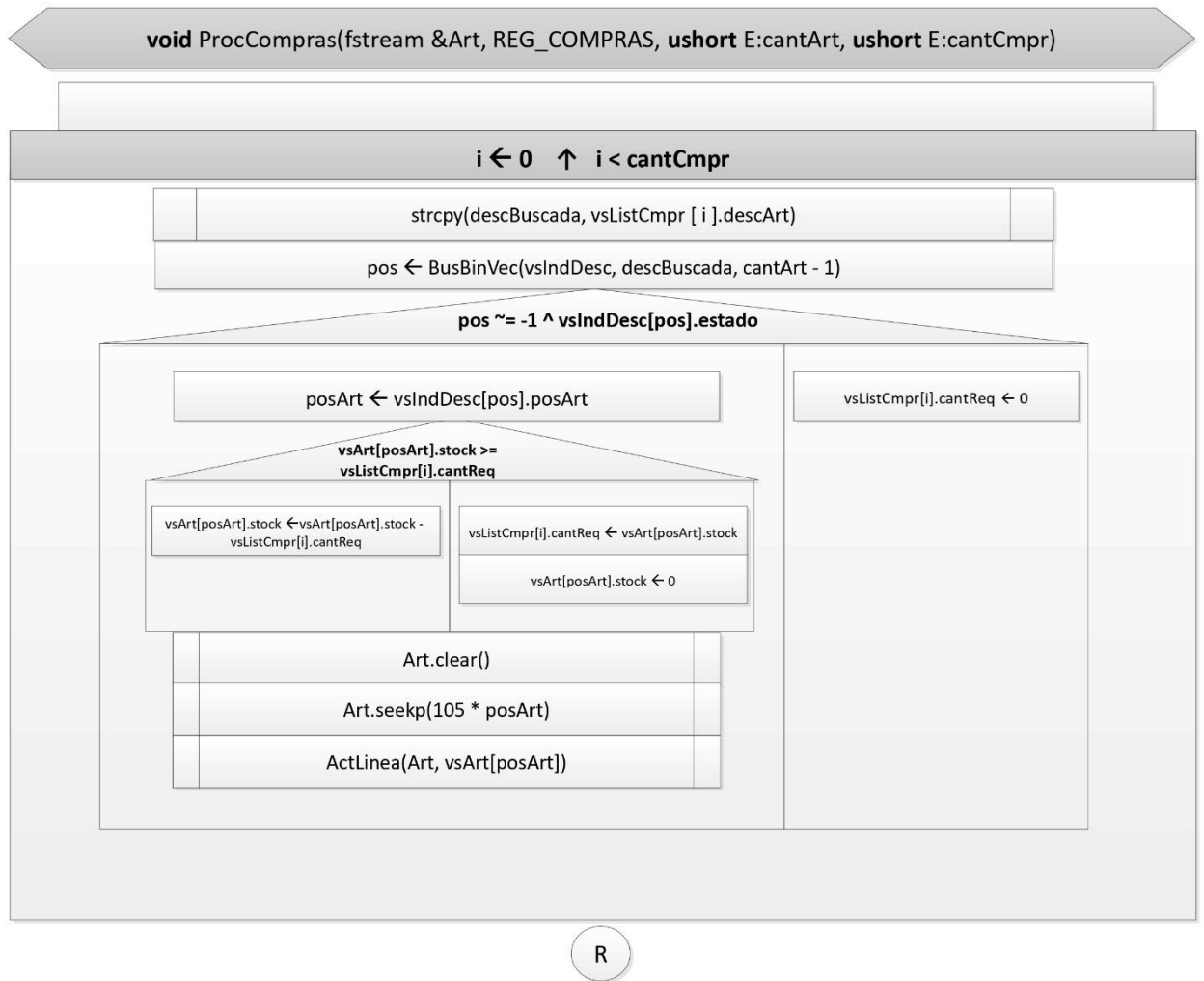
R



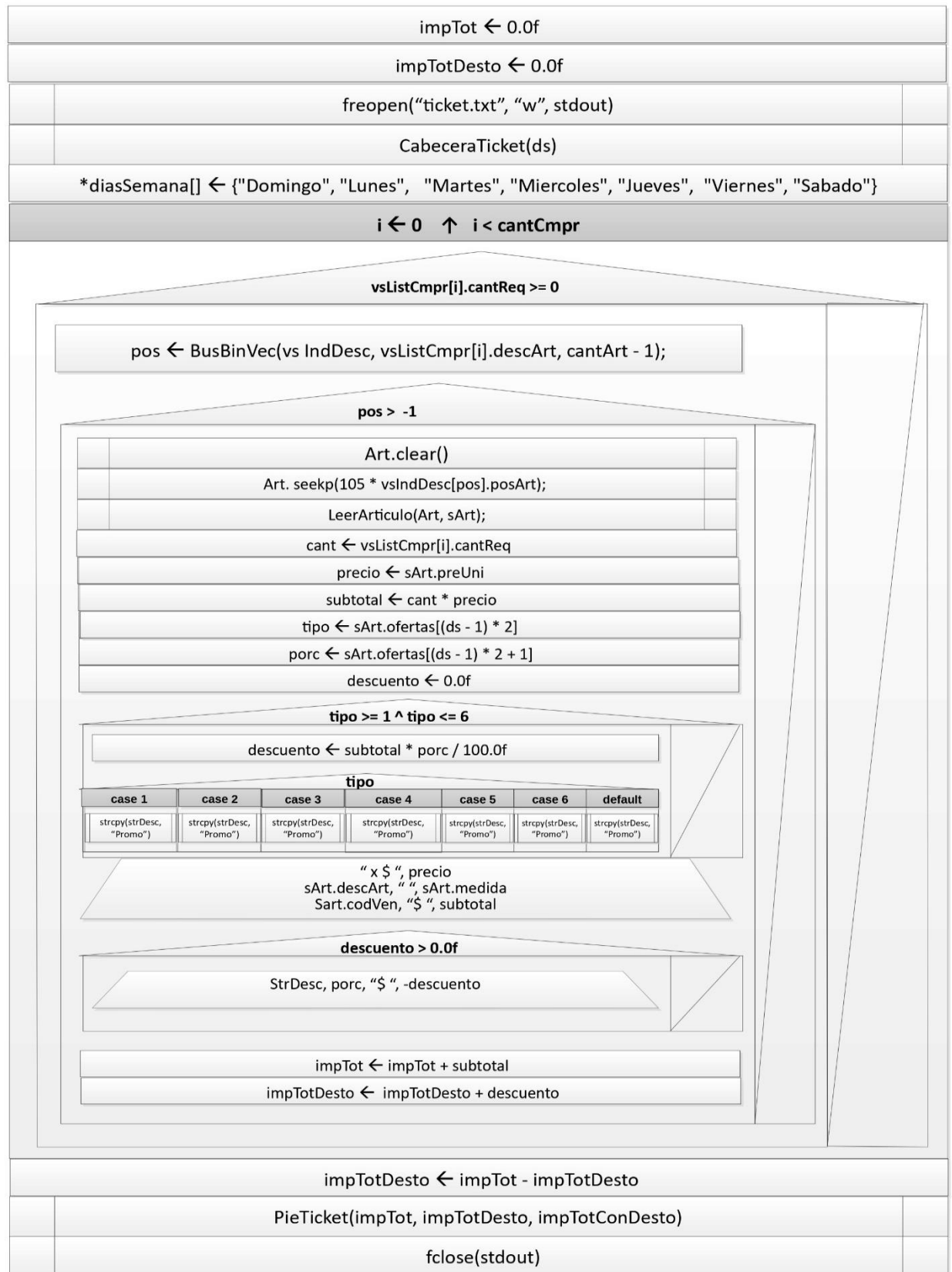


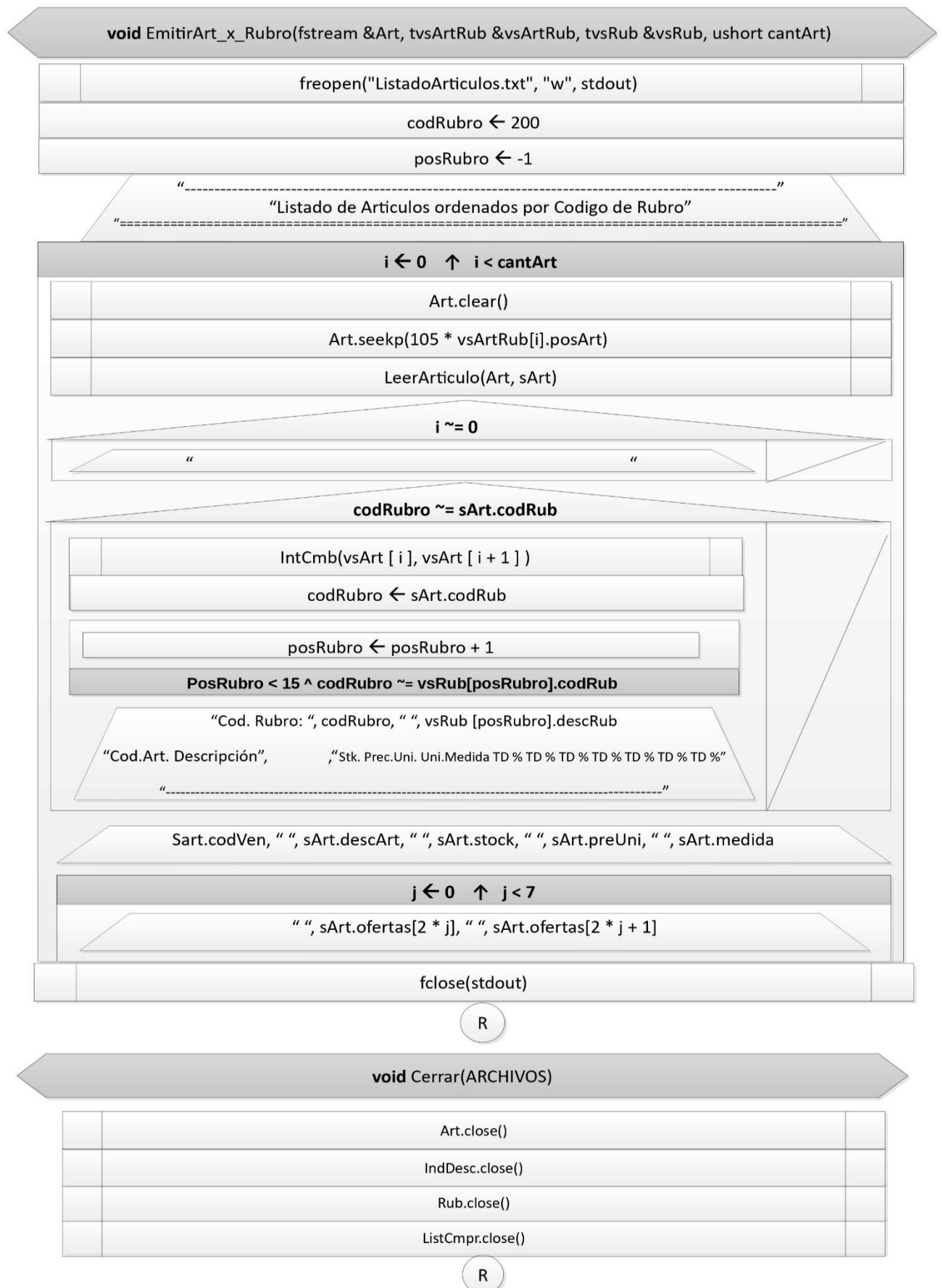






void EmitirTicket(fstream &Art, tvsIndDesc &vsIndDesc, tvsListCmpr &vsListCmpr, ushort cantArt, ushort cantCmpr)





Código

```
001: /*
002:  - Nombre del programa: TP1V1_K1021G10_HOLM FEDERICO.cpp
003:  - fecha entrega: 06/08/2025
004:  - Nro. versión: 1
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 LeerDescripcion(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 LeerDescripcion(ifstream &IndDesc, tsIndDesc &sIndDesc) {
173:     IndDesc.get(sIndDesc.descArt, 31);
174:     IndDesc >> sIndDesc.posArt >> sIndDesc.estado;
175:     IndDesc.ignore(2, '\n');
176:     return IndDesc.good();
177: } // LeerDescripcion
178:
179: bool LeerRubro(ifstream &Rub, tsRub &sRub) {
180:     Rub >> sRub.codRub;
181:     Rub.ignore();
182:     Rub.get(sRub.descRub, 21);
183:     Rub.ignore(2, '\n');
184:     return Rub.good();
185: } // LeerRubro
186:
187: bool LeerCompra(ifstream &ListCmpr, tsCompra &sCompra) {
188:     ListCmpr.get(sCompra.descArt, 31);
189:     ListCmpr >> sCompra.cantReq;
190:     ListCmpr.ignore(2, '\n');
191:     return ListCmpr.good();
192: } // LeerCompra
193:
194: void PieTicket(float impTot, float impTotDesto, float impTotConDesto) {
195:     float pagoUsuario = impTotConDesto; // El comprador paga exacto
196:
197:     float vuelto = pagoUsuario - impTotConDesto;
198:
199:     cout << setw(42) << left << "SubTot. sin descuentos....:" << "$ " << setw(10)
200:          << right << impTot << '\n'
201:          << setw(42) << left << "Descuentos por promociones:" << "$ " << setw(10)
202:          << right << -impTotDesto << '\n'
203:          << Replicate(' ', 54) << '\n'
204:          << setw(42) << left << "T O T A L" << "$ " << setw(10) << right
205:          << impTotConDesto << '\n'
206:          << Replicate(' ', 54) << '\n'
207:          << setw(42) << left << "Su pago con Mercado Pago:" << "$ " << setw(10)
208:          << right << pagoUsuario << endl
209:          << setw(42) << left << "Su vuelto:" << "$ " << setw(10) << right
210:          << vuelto << '\n'
211:          << Replicate(' ', 8) << "G R A C I A S P O R S U C O M P R A\n"
212:          << Replicate(' ', 8) << "Para consultas, sugerencias o reclamos\n"
```

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

```
285:     } else {
286:         li = pm + 1;
287:     }
288: }
289:
290: return -1; // No encontrado
291: } // BusBinVec
292:
293: string Replicate(char car, ushort n) {
294:     string resultado = "";
295:     for (ushort i = 0; i < n; i++)
296:         resultado += car;
297:     return resultado;
298: } // Replicate
299:
300: void Abrir(ARCHIVOS) {
301:     Art.open("Articulos.txt");
302:     IndDesc.open("IndDescripArt.txt");
303:     Rub.open("Rubros.txt");
304:     ListCmpr.open("ListaCompras.txt");
305: } // Abrir
306:
307: void VolcarArchivos(ARCHIVOS, REGISTROS, ushort &cantArt, ushort &cantCmpr) {
308:     tsArt sArt;
309:     tsIndDesc sIndDesc;
310:     tsRub sRub;
311:     tsCompra sCompra;
312:     cantArt = 0;
313:     cantCmpr = 0;
314:
315:     while (LeerArticulo(Art, sArt) && cantArt <= MAX_ART) {
316:         vsArtRub[cantArt].codRub = sArt.codRub;
317:         vsArtRub[cantArt].posArt = cantArt;
318:         cantArt++;
319:     }
320:     for (ushort i = 0; LeerDescripcion(IndDesc, sIndDesc) && i < cantArt; i++)
321:         vsIndDesc[i] = sIndDesc;
322:     for (ushort i = 0; LeerRubro(Rub, sRub) && i < CANT_RUB; i++)
323:         vsRub[i] = sRub;
324:     while (LeerCompra(ListCmpr, sCompra) && cantCmpr <= MAX_COMPRAS) {
325:         vsListCmpr[cantCmpr] = sCompra;
326:         cantCmpr++;
327:     }
328:
329:     OrdxBur(vsArtRub, cantArt);
330: } // VolcarArchivos
331:
332: void ProcCompras(fstream &Art, tvsIndDesc &vsIndDesc, tvsListCmpr &vsListCmpr,
333:     ushort cantArt, ushort cantCmpr) {
334:     str30 descBuscada;
335:     int pos;
336:     ushort posArt;
337:     tsArt sArt;
338:     Art << fixed << setprecision(2);
339:
340:     for (ushort i = 0; i < cantCmpr; i++) {
341:         strcpy(descBuscada, vsListCmpr[i].descArt);
342:         pos = BusBinVec(vsIndDesc, descBuscada, cantArt - 1);
343:
344:         if (pos != -1 && vsIndDesc[pos].estado) {
345:             posArt = vsIndDesc[pos].posArt;
346:
347:             Art.clear();
348:             Art.seekp(105 * posArt);
349:             LeerArticulo(Art, sArt);
350:
351:             if (sArt.stock >= vsListCmpr[i].cantReq) {
352:                 sArt.stock -= vsListCmpr[i].cantReq;
353:             } else {
354:                 vsListCmpr[i].cantReq = sArt.stock;
355:                 sArt.stock = 0;
356:             }
357:         }
358:     }
359: }
```



```
357:     Art.clear();
358:     Art.seekp(105 * posArt);
359:     ActLinea(Art, sArt);
360:
361: } else {
362:     vsListCmpr[i].cantReq = 0;
363: }
364: }
365: } // ProcCompras
366:
367: void EmitirTicket(fstream &Art, tvsIndDesc &vsIndDesc, tvsListCmpr &vsListCmpr,
368:     ushort cantArt, ushort cantCmpr) {
369:     int ds;
370:     float impTot = 0.0f, impTotDesto = 0.0f;
371:     tsArt sArt;
372:
373:     freopen("Ticket.txt", "w", stdout);
374:     CabeceraTicket(ds);
375:     cout << fixed << setprecision(2) << setfill(' ');
376:
377:     for (ushort i = 0; i < cantCmpr; i++) {
378:         if (vsListCmpr[i].cantReq >= 0) {
379:             int pos = BusBinVec(vsIndDesc, vsListCmpr[i].descArt, cantArt - 1);
380:             if (pos > -1) {
381:                 Art.clear();
382:                 Art.seekp(105 * vsIndDesc[pos].posArt);
383:                 LeerArticulo(Art, sArt);
384:
385:                 ushort cant = vsListCmpr[i].cantReq;
386:                 float precio = sArt.preUni;
387:                 float subtotal = cant * precio;
388:
389:                 ushort tipo = sArt.ofertas[(ds - 1) * 2];
390:                 ushort porc = sArt.ofertas[(ds - 1) * 2 + 1];
391:                 float descuento = 0.0f;
392:                 str10 strDesc;
393:
394:                 if (tipo >= 1 && tipo <= 6) // Solo aplicar si es válido
395:                     descuento = subtotal * porc / 100.0f;
396:
397:                 switch (tipo) {
398:                     case 1:
399:                         strcpy(strDesc, "Promo");
400:                         break;
401:                     case 2:
402:                         strcpy(strDesc, "Marca");
403:                         break;
404:                     case 3:
405:                         strcpy(strDesc, "Jub.");
406:                         break;
407:                     case 4:
408:                         strcpy(strDesc, "Comunid.");
409:                         break;
410:                     case 5:
411:                         strcpy(strDesc, "MercPago");
412:                         break;
413:                     case 6:
414:                         strcpy(strDesc, "ANSES");
415:                         break;
416:                     default:
417:                         strcpy(strDesc, "SinPromo");
418:                         break;
419:                 }
420:
421:                 cout << setw(2) << right << cant << " x $ " << setw(9) << precio << '\n'
422:                     << setw(30) << left << sArt.descArt << ' ' << setw(10)
423:                     << sArt.medida << '\n'
424:                     << setw(8) << right << sArt.codVen << setw(36) << "$ " << setw(10)
425:                     << subtotal << '\n';
426:
427:                 if (descuento > 0.0f) {
428:                     cout << setw(12) << left << strDesc << setw(5) << right << porc
```

23

Archivos de entrada

Articulos.txt:

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

Archivos de Salida

ListadoArticulos.txt:

----- Listado de Articulos ordenados porCodigo de Rubro -----												
Cod. Rubro: 1 Frutas												
Cod.Art.	Descripcion	Stk.	Prec.Uni.	Uni.Medida	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %
98024047	Banana Cavendish	0	1193.41	kg	6	58	6	13	6	13	3	8
75291495	Naranja Valencia	5661	2681.34	kg	1	36	6	23	6	4	6	94
85944160	Manzana Red Delicious	0	8922.24	kg	2	49	2	0	1	86	5	74
Cod. Rubro: 2 Verduras												
Cod.Art.	Descripcion	Stk.	Prec.Uni.	Uni.Medida	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %
90880352	Tomate Perita	1579	1575.23	kg	2	44	1	78	6	0	5	12
23880006	Lechuga Romana	315	3341.80	kg	5	7	0	96	6	19	5	80
65805349	Choclo	1830	9441.03	kg	3	22	5	53	0	15	3	31
Cod. Rubro: 3 Lácteos												
Cod.Art.	Descripcion	Stk.	Prec.Uni.	Uni.Medida	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %
75287095	Queso Mozzarella	2813	5897.99	l	5	15	3	43	2	77	1	59
3772978	Leche Entera	8566	6664.69	l	0	0	1	87	6	8	5	20
43812040	Yogur Natural	2367	2761.75	l	1	86	5	90	1	44	6	70
Cod. Rubro: 4 Carnes												
Cod.Art.	Descripcion	Stk.	Prec.Uni.	Uni.Medida	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %
99496819	Pechuga de Pollo	9221	5718.59	kg	2	33	2	56	2	63	3	5
30540322	Costilla de Cerdo	1935	2244.21	kg	5	41	5	91	6	1	1	88
81807170	Filet de Res	1798	8042.99	kg	4	45	1	57	5	4	2	4
Cod. Rubro: 5 Bebidas												
Cod.Art.	Descripcion	Stk.	Prec.Uni.	Uni.Medida	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %
79944212	Pepsi Max	4332	3035.10	l	1	66	6	61	2	10	2	93
13316861	Coca-Cola Zero	9640	7968.79	l	3	56	2	83	1	35	5	73
26304969	Agua Mineral	2494	4101.67	l	4	86	5	22	5	60	4	62
Cod. Rubro: 6 Pescados												
Cod.Art.	Descripcion	Stk.	Prec.Uni.	Uni.Medida	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %
96060718	Atun Fresco	3727	4384.19	kg	1	65	5	33	4	95	6	69
65758412	Salmon Fresco	8793	8609.23	kg	6	69	6	7	3	20	1	80
95959493	Merluza Congelada	6071	3422.22	kg	2	21	0	20	0	85	4	61

Cod. Rubro: 7 Panadería																				
Cod.Art. Descripcion	Stk.	Prec.Uni.	Uni.Medida	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %
68466733 Baguette Francesa	4982	5911.89	unidad	1	9	1	89	5	18	0	17	4	58	2	18	2	61			
33415557 Medialunas	6934	847.05	unidad	0	32	3	90	2	33	3	20	2	84	0	82	1	96			
49249004 Pan Integral	6145	3504.18	unidad	5	93	3	60	4	49	4	90	6	19	6	95	3	25			

Cod. Rubro: 8 Higiene																				
Cod.Art. Descripcion	Stk.	Prec.Uni.	Uni.Medida	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %
10840510 Papel Higienico	5100	3415.63	unidad	1	56	2	2	2	79	4	82	6	4	3	55	0	81			
28853710 Shampoo	8484	6525.76	unidad	1	30	0	71	4	75	3	58	5	10	2	21	6	70			
766444 Jabon Liquido	2911	9198.66	unidad	2	74	2	29	6	48	6	35	3	54	5	85	4	99			

Cod. Rubro: 9 Limpieza																				
Cod.Art. Descripcion	Stk.	Prec.Uni.	Uni.Medida	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %
18565176 Suavizante	6727	2687.28	unidad	0	97	2	30	6	36	2	74	4	77	3	83	0	31			
57285423 Limpiador Multiusos	9867	1029.55	unidad	0	27	2	17	1	86	6	9	6	80	1	80	2	48			
82574849 Detergente Polvo	2055	6237.69	unidad	1	83	1	61	2	98	4	55	5	84	4	38	4	34			

Cod. Rubro: 10 Dulces																				
Cod.Art. Descripcion	Stk.	Prec.Uni.	Uni.Medida	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %
77811558 Gomititas Frutales	9419	3692.72	unidad	6	84	0	96	5	91	6	22	6	87	1	86	0	37			
29319846 Chocolate con Leche	4561	8125.42	unidad	6	54	3	65	6	45	1	28	2	72	2	61	0	34			
99840726 Caramelos Surtidos	28	9205.07	unidad	5	55	5	75	3	63	3	30	6	45	0	56	3	98			

Cod. Rubro: 11 Cereales																				
Cod.Art. Descripcion	Stk.	Prec.Uni.	Uni.Medida	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %
19338712 Corn Flakes	7522	4249.51	kg	1	30	0	52	2	36	3	2	5	80	6	39	2	58			
69233733 Granola	1392	5755.99	kg	3	95	6	53	3	35	5	9	6	26	2	62	6	48			
90991857 Avena	6306	5676.29	kg	0	24	2	2	4	80	3	13	1	57	3	32	3	60			

Cod. Rubro: 12 Salsas																				
Cod.Art. Descripcion	Stk.	Prec.Uni.	Uni.Medida	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %
79832716 Salsa de Tomate	1888	6937.68	unidad	4	57	4	52	5	2	5	73	0	5	5	82	0	44			
44255999 Mostaza	6185	399.88	unidad	4	76	2	33	4	72	1	39	2	47	1	72	5	81			
37897727 Mayonesa	1989	3664.18	unidad	0	53	0	50	3	34	6	16	6	19	1	53	2	52			

Cod. Rubro: 13 Congelados																				
Cod.Art. Descripcion	Stk.	Prec.Uni.	Uni.Medida	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %
80786032 Helado Vainilla	3499	9071.57	unidad	4	14	0	64	0	7	3	61	2	80	4	34	2	74			
24358058 Pizza Congelada	8943	3279.19	unidad	2	19	4	70	2	70	1	36	6	49	0	77	5	41			
6643982 Verduras Congeladas	4696	5204.42	unidad	4	94	0	36	3	25	5	45	2	83	3	4	5	5			

Cod. Rubro: 14 Aceites																				
Cod.Art. Descripcion	Stk.	Prec.Uni.	Uni.Medida	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %
71843253 Aceite de Coco	7115	1055.04	l	4	72	3	51	3	80	1	64	0	41	1	10	3	30			
83271845 Aceite de Oliva	6599	3005.36	l	6	75	0	82	6	92	2	82	4	44	3	78	2	13			
57622019 Aceite de Girasol	4792	6645.97	l	2	64	4	73	4	34	6	88	3	18	1	45	0	64			

Cod. Rubro: 15 Enlatados																				
Cod.Art. Descripcion	Stk.	Prec.Uni.	Uni.Medida	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %	TD %
15742093 Atun en Aceite	4262	6370.83	unidad	3	29	3	22	2	20	1	8	1	36	0	25	5	96			
19078262 Arvejas	2326	7332.64	unidad	3	70	0	54	3	40	3	46	3	64	3	79	4	89			
2384353 Maiz Dulce	2154	9349.45	unidad	0	82	6	89	2	96	6	31	6	35	5	76	1	34			

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

98024047	1	Banana Cavendish	0	1193.41	kg	6	58	6	13	6	13	3	8	5	80	4	98	2	15	
79832716	12	Salsa de Tomate	1888	6937.68	unidad	4	57	4	52	5	2	5	73	0	5	5	82	0	44	
44255999	12	Mostaza	6185	399.88	unidad	4	76	2	33	4	72	1	39	2	47	1	72	5	81	
77811558	10	Gomititas Frutales	9419	3692.72	unidad	6	84	0	96	5	91	6	22	6	87	1	86	0	37	
71843253	14	Aceite de Coco	7115	1055.04	l	4	72	3	51	3	80	1	64	0	41	1	10	3	30	

79944212	5	Pepsi Max	4332	3035.10 l	1 66 6 61 2 10 2 93 3 29 3 81 4 7
15742093	15	Atun en Aceite	4262	6370.83 unidad	3 29 3 22 2 20 1 8 1 36 0 25 5 96
75287095	3	Queso Mozzarella	2813	5897.99 l	5 15 3 43 2 77 1 59 1 53 6 39 2 47
96060718	6	Atun Fresco	3727	4384.19 kg	1 65 5 33 4 95 6 69 1 25 3 54 5 44
19338712	11	Corn Flakes	7522	4249.51 kg	1 30 0 52 2 36 3 2 5 80 6 39 2 58
10840510	8	Papel Higienico	5100	3415.63 unidad	1 56 2 2 2 79 4 82 6 4 3 55 0 81
99496819	4	Pechuga de Pollo	9221	5718.59 kg	2 33 2 56 2 63 3 5 3 34 4 87 2 12
90880352	2	Tomate Perita	1579	1575.23 kg	2 44 1 78 6 0 5 12 6 62 2 88 2 38
65758412	6	Salmon Fresco	8793	8609.23 kg	6 69 6 7 3 20 1 80 4 10 2 12 1 6
28853710	8	Shampoo	8484	6525.76 unidad	1 30 0 71 4 75 3 58 5 10 2 21 6 70
68466733	7	Baguette Francesa	4982	5911.89 unidad	1 9 1 89 5 18 0 17 4 58 2 18 2 61
29319846	10	Chocolate con Leche	4561	8125.42 unidad	6 54 3 65 6 45 1 28 2 72 2 61 0 34
30540322	4	Costilla de Cerdo	1935	2244.21 kg	5 41 5 91 6 1 1 88 5 32 6 15 0 68
19078262	15	Arvejas	2326	7332.64 unidad	3 70 0 54 3 40 3 46 3 64 3 79 4 89
81807170	4	Filet de Res	1798	8042.99 kg	4 45 1 57 5 4 2 4 1 11 1 52 1 11
3772978	3	Leche Entera	8566	6664.69 l	0 0 1 87 6 8 5 20 4 89 5 99 2 99
99840726	10	Caramelos Surtidos	28	9205.07 unidad	5 55 5 75 3 63 3 30 6 45 0 56 3 98
23880006	2	Lechuga Romana	315	3341.80 kg	5 7 0 96 6 19 5 80 2 25 1 52 4 26
83271845	14	Aceite de Oliva	6599	3005.36 l	6 75 0 82 6 92 2 82 4 44 3 78 2 13
18565176	9	Suavizante	6727	2687.28 unidad	0 97 2 30 6 36 2 74 4 77 3 83 0 31
57622019	14	Aceite de Girasol	4792	6645.97 l	2 64 4 73 4 34 6 88 3 18 1 45 0 64
75291495	1	Naranja Valencia	5661	2681.34 kg	1 36 6 23 6 4 6 94 2 67 5 73 0 76
65805349	2	Choclo	1830	9441.03 kg	3 22 5 53 0 15 3 31 1 77 0 23 0 90
95959493	6	Merluza Congelada	6071	3422.22 kg	2 21 0 20 0 85 4 61 3 13 2 61 1 40
85944160	1	Manzana Red Delicious	0	8922.24 kg	2 49 2 0 1 86 5 74 4 10 1 17 2 45
2384353	15	Maiz Dulce	2154	9349.45 unidad	0 82 6 89 2 96 6 31 6 35 5 76 1 34
69233733	11	Granola	1392	5755.99 kg	3 95 6 53 3 35 5 9 6 26 2 62 6 48
57285423	9	Limpiador Multiusos	9867	1029.55 unidad	0 27 2 17 1 86 6 9 6 80 1 80 2 48
13316861	5	Coca-Cola Zero	9640	7968.79 l	3 56 2 83 1 35 5 73 4 73 3 26 2 48
80786032	13	Helado Vainilla	3499	9071.57 unidad	4 14 0 64 0 7 3 61 2 80 4 34 2 74
37897727	12	Mayonesa	1989	3664.18 unidad	0 53 0 50 3 34 6 16 6 19 1 53 2 52
82574849	9	Detergente Polvo	2055	6237.69 unidad	1 83 1 61 2 98 4 55 5 84 4 38 4 34
33415557	7	Medialunas	6934	847.05 unidad	0 32 3 90 2 33 3 20 2 84 0 82 1 96
766444	8	Jabon Liquido	2911	9198.66 unidad	2 74 2 29 6 48 6 35 3 54 5 85 4 99
24358058	13	Pizza Congelada	8943	3279.19 unidad	2 19 4 70 2 70 1 36 6 49 0 77 5 41
90991857	11	Avena	6306	5676.29 kg	0 24 2 2 4 80 3 13 1 57 3 32 3 60
49249004	7	Pan Integral	6145	3504.18 unidad	5 93 3 60 4 49 4 90 6 19 6 95 3 25
26304969	5	Agua Mineral	2494	4101.67 l	4 86 5 22 5 60 4 62 0 76 5 49 5 57
43812040	3	Yogur Natural	2367	2761.75 l	1 86 5 90 1 44 6 70 1 2 4 74 4 23
6643982	13	Verduras Congeladas	4696	5204.42 unidad	4 94 0 36 3 25 5 45 2 83 3 4 5 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: Lunes 04/08/2025
 Hora: 19:08:46
 Nro. Ticket: 9999-99999999
 Nro. Caja: 9999

 F A C T U R A - B
 ORIGINAL

 0 x \$ 1193.41
 Banana Cavendish kg \$ 0.00
 98024047

 5 x \$ 6937.68
 Salsa de Tomate unidad \$ 34688.40
 79832716 \$ -18037.97
 Comunid. 52

 10 x \$ 399.88
 Mostaza unidad \$ 3998.80
 44255999 \$ -1319.60
 Marca 33

20 x \$ 3692.72 Gomitas Frutales 77811558	unidad	\$ 73854.40
8 x \$ 1055.04 Aceite de Coco 71843253 Jub. 51	1	\$ 8440.32 \$ -4304.56
15 x \$ 3035.10 Pepsi Max 79944212 ANSES 61	1	\$ 45526.50 \$ -27771.16
0 x \$ 6370.83 Atun en Aceite 15742093	unidad	\$ 0.00
7 x \$ 5897.99 Queso Mozzarella 75287095 Jub. 43	1	\$ 41285.93 \$ -17752.95
0 x \$ 4384.19 Atun Fresco 96060718	kg	\$ 0.00
13 x \$ 4249.51 Corn Flakes 19338712	kg	\$ 55243.62
9 x \$ 3415.63 Papel Higienico 10840510 Marca 2	unidad	\$ 30740.67 \$ -614.81
4 x \$ 5718.59 Pechuga de Pollo 99496819 Marca 56	kg	\$ 22874.36 \$ -12809.64
0 x \$ 1575.23 Tomate Perita 90880352	kg	\$ 0.00
10 x \$ 8609.23 Salmon Fresco 65758412 ANSES 7	kg	\$ 86092.30 \$ -6026.46
12 x \$ 6525.76 Shampoo 28853710	unidad	\$ 78309.12
3 x \$ 5911.89 Baguette Francesa 68466733 Promo 89	unidad	\$ 17735.67 \$ -15784.75
8 x \$ 8125.42 Chocolate con Leche 29319846 Jub. 65	unidad	\$ 65003.36 \$ -42252.18
5 x \$ 2244.21 Costilla de Cerdo 30540322 MercPago 91	kg	\$ 11221.05 \$ -10211.16
17 x \$ 7332.64 Arvejas	unidad	

19078262		\$	124654.88
16 x \$ 8042.99			
Filet de Res	kg		
81807170		\$	128687.84
Promo 57		\$	-73352.07
0 x \$ 6664.69			
Leche Entera	l		
3772978		\$	0.00
19 x \$ 9205.07			
Caramelos Surtidos	unidad		
99840726		\$	174896.34
MercPago 75		\$	-131172.25
11 x \$ 3341.80			
Lechuga Romana	kg		
23880006		\$	36759.80
15 x \$ 3005.36			
Aceite de Oliva	l		
83271845		\$	45080.40
10 x \$ 2687.28			
Suavizante	unidad		
18565176		\$	26872.80
Marca 30		\$	-8061.84
0 x \$ 6645.97			
Aceite de Girasol	l		
57622019		\$	0.00
6 x \$ 2681.34			
Naranja Valencia	kg		
75291495		\$	16088.04
ANSES 23		\$	-3700.25
0 x \$ 9441.03			
Choclo	kg		
65805349		\$	0.00
0 x \$ 3422.22			
Merluza Congelada	kg		
95959493		\$	0.00
5 x \$ 8922.24			
Manzana Red Delicious	kg		
85944160		\$	44611.20
SubTot. sin descuentos....:		\$	1172665.88
Descuentos por promociones:		\$	-373171.66
=====			
T O T A L		\$	799494.25
=====			
Su pago con Mercado Pago:		\$	799494.25
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