
# *Elementos de Programación*

## *UNIDAD 7. ARRAYS*

## *Anexo Ejercicios Resueltos*

Una empresa tiene 10 vendedores identificados con un código numérico que va del 101 al 110. La empresa comercializa productos que son codificados con números de 4 cifras. No se sabe la cantidad exacta de productos que vende la empresa, pero sí se sabe que no son más de 30.

Se desea realizar un programa que permita:

1. Ingresar los códigos de producto que comercializa la empresa junto con su precio. La carga finaliza con un código igual a 10000.
2. Ingresar las ventas realizadas. Para ello por cada factura se ingresa:
    a. Código de producto
    b. Código de vendedor
    c. Cantidad de unidades vendidas

    Las ventas finalizan con un código de vendedor igual a 0.
3. Al finalizar mostrar el detalle de los importes vendidos por cada vendedor de cada uno de los productos.
4. Mostrar el o los vendedores que realizaron un mayor importe de ventas
5. Indicar aquellos productos que no fueron vendidos por ningún vendedor.

Para encarar un ejercicio de este tipo primero se debe analizar y planificar una estrategia de resolución. El punto 1 nos pide ingresar los códigos de los productos y el precio. Nos dice que no se sabe la cantidad de datos, pero sí que no son más de 30, por lo tanto, al reservar memoria lo haremos por el máximo posible es decir de 30. Como tenemos que almacenar 2 datos, el código y el precio vamos a necesitar dos vectores que estén en paralelo, es decir que en la posición 0 del vector de código vamos a almacenar el código del producto y al mismo tiempo en la posición 0 del vector de precios vamos a almacenar el precio de dicho producto. Como no sabemos la cantidad exacta de datos la función de carga DEBE retornar la cantidad de datos ingresados para luego poder usar ese dato en el resto de las funciones.

En el punto 2 se procesan las ventas, para ello se ingresa el código de producto y el código de vendedor. Como se debe almacenar el detalle de importes de ventas de cada vendedor por cada producto, vamos a necesitar una matriz. Debido a que dicha matriz guarda importes va a ser una matriz de float. Dicha matriz va a tener una columna por cada vendedor (son 10 en total) y una fila por cada producto (máximo 30) entonces se debe declarar como de 30 x 10 por más que luego tengamos filas que no se utilicen. Dicha matriz como va a acumular debe estar inicializada en 0.

La información de las facturas nos permitirá posicionarnos en matriz para acumular el dato:

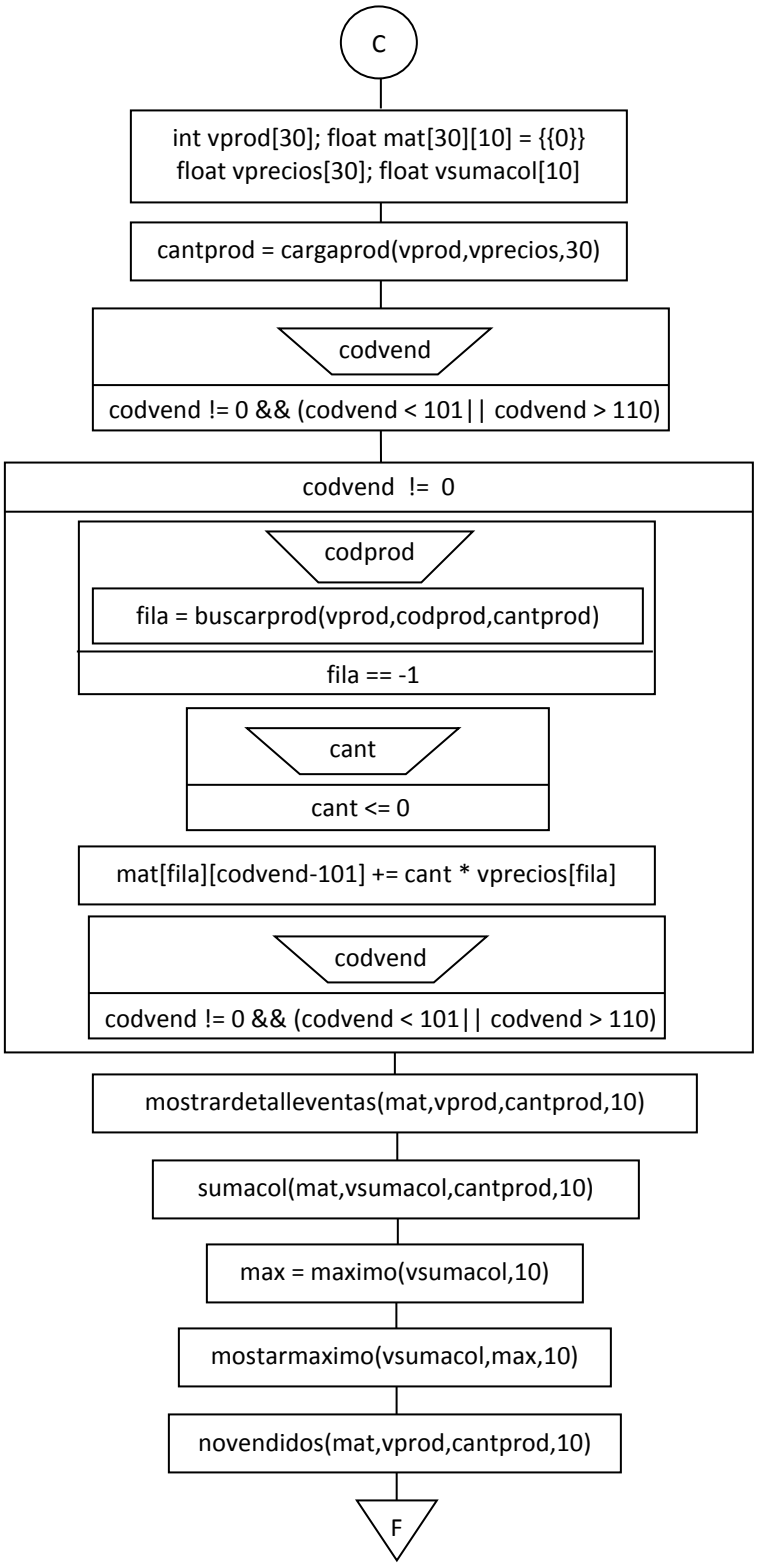
- La columna será el código de vendedor que como son correlativos de 101 a 110 y nuestra matriz comienza en 0 entonces tendremos que restarle 101 al código ingresado para posicionarnos en la columna correspondiente.
- La fila la da el código de producto, pero como los códigos se ingresaron previamente por teclado para saber a qué fila de la matriz corresponde, se debe realizar una búsqueda sobre el vector de códigos y recuperar la posición donde se encuentra dicho código. Esa posición nos indica la fila de la matriz
- El dato para acumular no es la cantidad sino el importe, por lo que se debe recurrir al vector de precios con el subíndice de donde encontramos el producto. Dicho precio se multiplica por la cantidad ingresada y ese importe se acumula en la matriz.

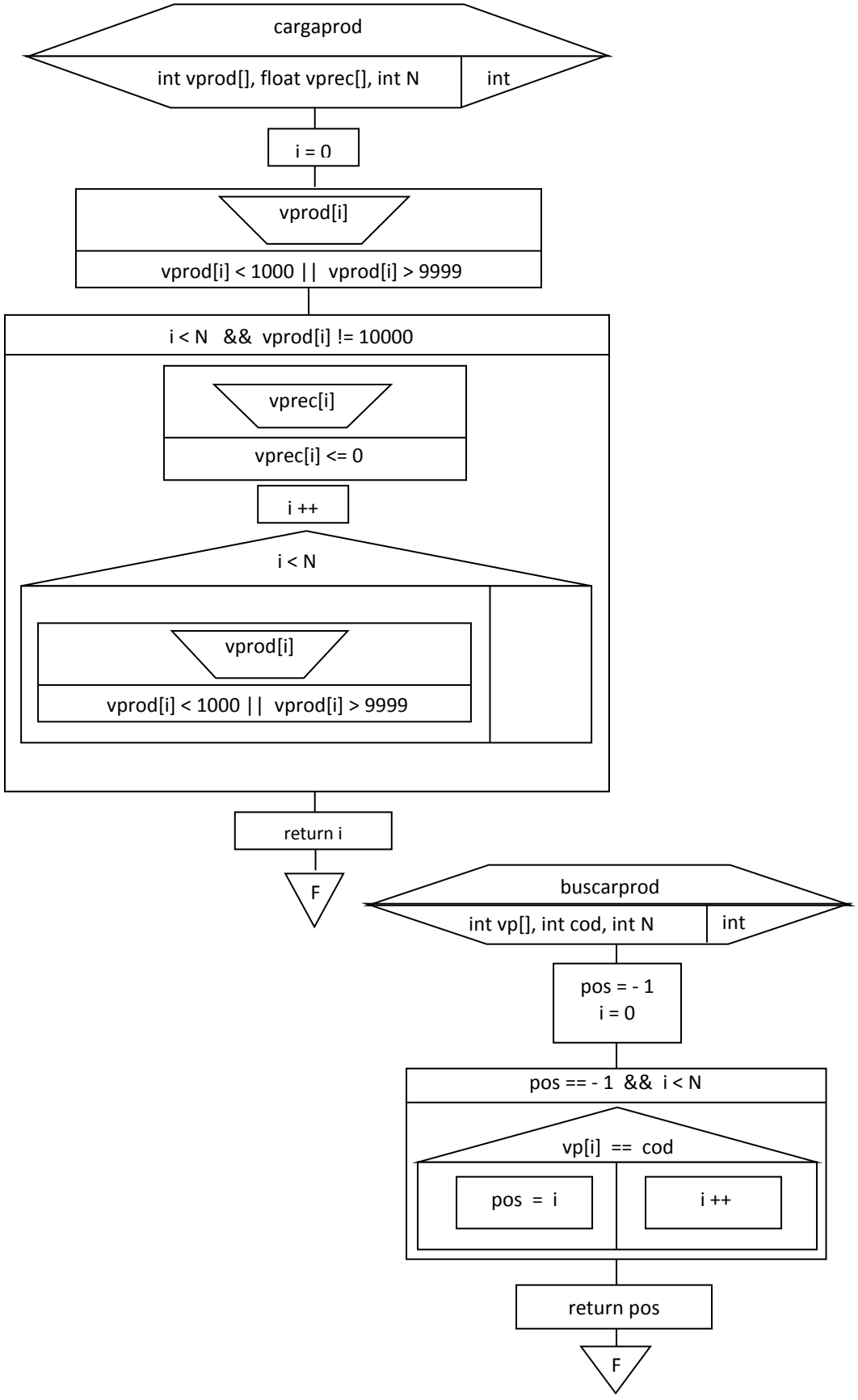
Una vez finalizado el ingreso de facturas (cuando se ingresa un vendedor con código igual a 0) se comienzan a resolver los informes que nos solicitan.

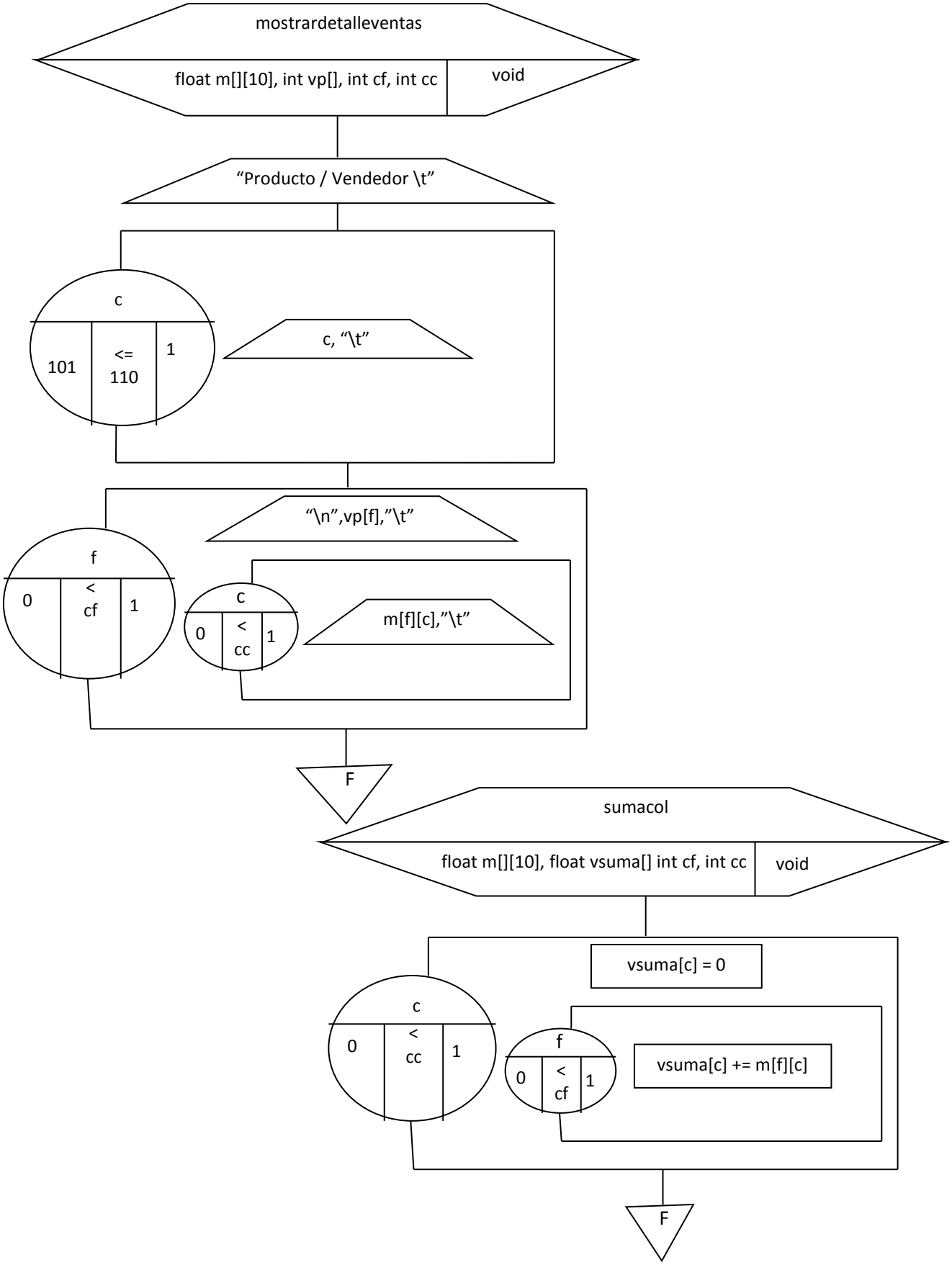
El punto 3 nos pide el detalle ventas, mostrando importe vendido por cada vendedor de cada producto por lo que se debe mostrar toda la matriz encolumnada y con los títulos de filas y columnas correspondientes.

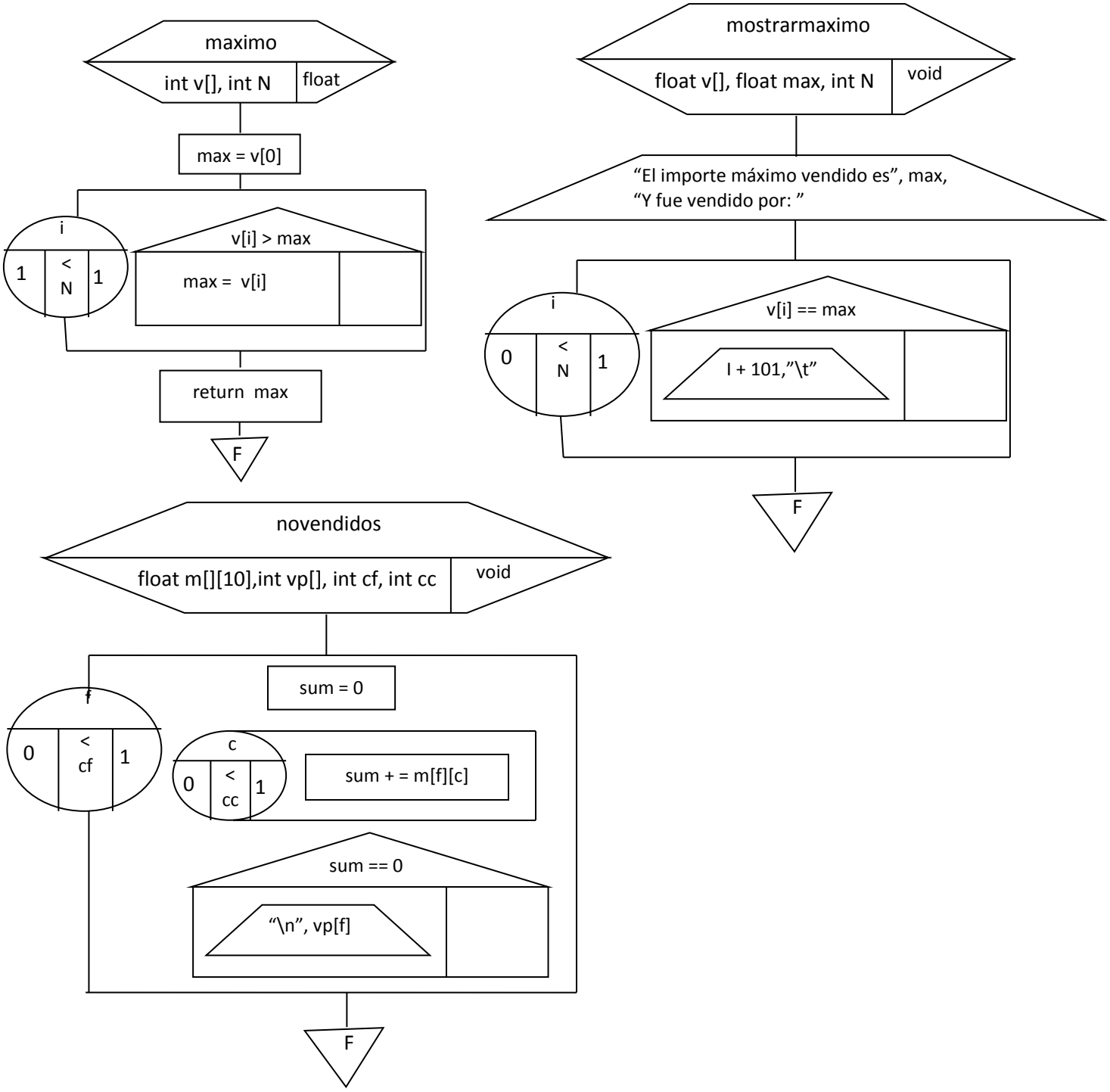
El punto 4 solicita mostrar el o los vendedores que realizaron un mayor importe de ventas. En la matriz la información de los importes de venta de cada vendedor esta detallada por producto, pero en este punto nos pide el total por lo que será necesario sumar todos los importes vendidos para cada vendedor. Debemos realizar una suma por columna almacenando los resultados en un vector. Luego sobre ese vector se debe calcular el importe máximo y por último volver a recorrer ese vector para ver si ese importe máximo se repite o no en varios vendedores.

El punto 5 pide indicar aquellos productos que no fueron vendidos por ningún vendedor. La información de las ventas de cada producto se puede obtener recorriendo por fila la matriz, si sumamos toda una fila y esa suma es igual a 0, significa que no se vendió dicho producto.


C
int vprod[30]; float mat[30][10] = {{0}}
float vprecios[30]; float vsumacol[10]
cantprod = cargaprod(vprod,vprecios,30)
codvend
codvend != 0 && (codvend < 101|| codvend > 110)
codvend != 0
codprod
fila = buscarprod(vprod,codprod,cantprod)
fila == -1
cant
cant <= 0
mat[fila][codvend-101] += cant * vprecios[fila]
codvend
codvend != 0 && (codvend < 101|| codvend > 110)
mostrardetalleventas(mat,vprod,cantprod,10)
sumacol(mat,vsumacol,cantprod,10)
max = maximo(vsumacol,10)
mostarmaximo(vsumacol,max,10)
novendidos(mat,vprod,cantprod,10)
F


cargaprod
int vprod[], float vprec[], int N
int
i = 0
vprod[i]
vprod[i] < 1000 || vprod[i] > 9999
i < N && vprod[i] != 10000
vprec[i]
vprec[i] <= 0
i ++
i < N
vprod[i]
vprod[i] < 1000 || vprod[i] > 9999
return i
F
buscarprod
int vp[], int cod, int N
int
pos = - 1
i = 0
pos == - 1 && i < N
vp[i] == cod
pos = i
i ++
return pos
F


mostrardetalleventas
float m[][10], int vp[], int cf, int cc
void
“Producto / Vendedor \t”
c
101
<=
110
1
c, “\t”
“\n”,vp[f],”\t”
f
0
<
cf
1
c
0
<
cc
1
m[f][c],”\t”
F
sumacol
float m[][10], float vsuma[] int cf, int cc
void
vsuma[c] = 0
c
0
<
cc
1
f
0
<
cf
1
vsuma[c] += m[f][c]
F


maximo
int v[], int N
float
max = v[0]
i
1
<
N
1
v[i] > max
max = v[i]
return max
F
mostrarmaximo
float v[], float max, int N
void
“El importe máximo vendido es”, max,
“Y fue vendido por: ”
i
0
<
N
1
v[i] == max
I + 101,”\t”
F
novendidos
float m[][10],int vp[], int cf, int cc
void
sum = 0
f
0
<
cf
1
c
0
<
cc
1
sum + = m[f][c]
sum == 0
“\n”, vp[f]
F

```
#include <stdio.h>
int CargaProd (int[], float[], int);
int BuscarProd (int[], int, int);
void MostrarDetalleVtas (float[][10], int [], int, int);
void SumaCol (float[][10], float[], int, int);
float Maximo (float[], int);
void MostrarMaximo (float[], float, int);
void NoVendidos(float[][10], int[], int, int);

int main()
{
    int vprod[30], cantprod, codvend, codprod, fila, cant;
    float mat[30][10]={{0}}, vprecios[30], vsumacol[10], max;
    cantprod = CargaProd(vprod, vprecios, 30);
    do
    {
        printf("Ingrese el codigo del vendedor (entre 101 y 110, 0 para terminar):");
        scanf("%d",&codvend);
    }while(codvend != 0 && (codvend<101||codvend>110));

    while(codvend!=0)
    {
        do
        {
            printf("Ingrese el codigo de producto:");
            scanf("%d", &codprod);
            fila = BuscarProd(vprod,codprod, cantprod);
        }while(fila == -1);

        do
        {
            printf("Ingrese la cantidad vendida:");
            scanf("%d", &cant);
        }while (cant<=0);

        mat[fila][codvend-101]+=cant*vprecios[fila];

        do
        {
            printf("Ingrese el codigo del vendedor (entre 101 y 110, 0 para terminar):");
            scanf("%d",&codvend);
        }while(codvend != 0 && (codvend<101||codvend>110));
    }
    MostrarDetalleVtas(mat,vprod,cantprod,10);
    SumaCol(mat,vsumacol, cantprod,10);
    max=Maximo(vsumacol,10);
    MostrarMaximo(vsumacol, max,10);
    NoVendidos(mat,vprod,cantprod,10);
    return 0;
}
int CargaProd (int vprod[], float vprec[], int n)
{
    int i=0;
    do
    {
        printf("Ingrese el codigo de producto (4 cifras) :");
        scanf("%d", &vprod[i]);
    }while(vprod[i]<1000||vprod[i]>9999); //con 9999 obligo a ingresar al menos un producto

    while (i<n && vprod[i]!=10000)
    {
        do
        {
            printf("Ingrese el precio para el producto %d:",vprod[i]);
            scanf("%f", &vprec[i]);
        }while(vprec[i]<=0);

        i++;
        if (i<n)
            do
            {
                printf("Ingrese el codigo de producto (10000 para terminar): ");
```

```
                scanf("%d", &vprod[i]);
            }while(vprod[i]<1000||vprod[i]>10000);
    }
    return i;
}
int BuscarProd (int vp[], int cod, int n)
{
    int pos =-1, i=0;
    while (pos==-1 && i<n)
    {
        if (vp[i]==cod)
            pos=i;
        else
            i++;
    }
    return pos;
}
void MostrarDetalleVtas (float m[][10], int vp[], int cf, int cc)
{
    int c,f;
    printf("producto/vendedor");
    for (c=101;c<=110;c++)
        printf ("%6d", c);
    for (f=0;f<cf;f++)
    {
        printf("\n%17d",vp[f]);
        for (c=0;c<cc;c++)
            printf("%8.2f",m[f][c]);
    }
}
void SumaCol (float m[][10], float vsuma[], int cf, int cc)
{
    int f,c;
    for (c=0;c<cc;c++)
    {
        vsuma[c]=0;
        for(f=0;f<cf;f++)
            vsuma[c]+=m[f][c];
    }
}
float Maximo (float v[] , int n)
{
    float max = v[0];
    int i;
    for (i=1;i<n;i++)
        if (v[i]>max)
            max =v[i];
    return max;
}
void MostrarMaximo (float v[], float max, int n)
{
    int i;
    printf ("\nEl importe maximo vendido es: %5.2f y fue vendido por: ", max);
    for (i=0;i<n;i++)
        if (v[i]==max)
            printf ("%4d", i+101);
}
void NoVendidos(float m[][10], int vp[], int cf, int cc)
{
    int f,c;
    float sum;
    for (f=0;f<cf;f++)
    {
        sum=0;
        for (c=0;c<cc;c++)
            sum+=m[f][c];
        if (sum ==0)
            printf ("\nEl producto %d no fue vendido",vp[f]);
    }
}
```