Capítulo 8

8.6 Ejercicios resueltos

Ejercicio 8.6.1

Elaborar un algoritmo que maneje un sistema de control de obreros, que permita crear, hacer altas, bajas y cambios a un archivo de obreros y otro de plantas. El archivo de obreros contiene los datos de todos los obreros de la compañía, la cual se divide en varias plantas, es por ello que cada obrero tiene un dato que es la clave de la planta a la que pertenece. El archivo de plantas contiene los nombres de todas las plantas que conforman la compañía.

Obreros	Plantas
numero	clavePlanta
nombre	nombrePlanta
clavePlanta	
produc[5]	

Donde:

numero	Es el número de identificación del obrero.
nombre	Es el nombre del obrero.
clavePlanta	Es la clave de identificación de la planta.
produc[5]	Es un arreglo con 5 elementos: uno para almacenar la producción que hizo el obrero en cada uno de los 5 días de la semana.
nombrePlanta	Es el nombre de la planta en la que labora el obrero.

La captura de la producción de los obreros se debe hacer por día, es decir, se captura la producción del día 1 para todos los obreros, luego el día 2 para todos los obreros, y así sucesivamente hasta el día 5.

El algoritmo también debe proporcionar la posibilidad de hacer consultas de plantas y de obreros con dar la clave o el número respectivamente; además debe emitir los siguientes reportes:

```
                    REPORTE SEMANAL DE PRODUCCION
NUMERO NOMBRE        DIA1  DIA2  DIA3  DIA4  DIA5 PRO.SEMANA
-------------------------------------------------------------
999    XXXXXXXXXXXX   99    99    99    99    99     99
999    XXXXXXXXXXXX   99    99    99    99    99     99
            --
            --
            --
TOTAL  99 OBREROS     99    99    99    99    99     99
```

DIA 1, DIA2, DIA 3, DIA 4, DIA 5 son la producción que hizo el obrero en cada uno de los días correspondientes.
PRO. SEMANA es la sumatoria de la producción de los cinco días para obtener la producción semanal.
TOTAL es el total de obreros, y el total de la producción de todos los obreros por cada uno de los siete días y el total de producción de todos los obreros.

```
                        REPORTE DE PRODUCCION POR PLANTA
NUMERO NOMBRE          PLANTA  DIA1  DIA2  DIA3  DIA4  DIA5  PRO.SEMANA
-------------------------------------------------------------------------

  999  XXXXXXXXXXXX     99     99    99    99    99    99       99
  999  XXXXXXXXXXXX     99     99    99    99    99    99       99
       --
       --
  999  XXXXXXXXXXXX     99     99    99    99    99    99       99

TOTAL PLANTA  XXXXXXXXXXXXXXXX
       99 OBREROS              99    99    99    99    99       99
OBRERO MAS PRODUCTIVO:  XXXXXXXXXXXXXXXXXXXX
    .
    .
TOTAL PLANTA  XXXXXXXXXXXXXXXX
       99 OBREROS              99    99    99    99    99       99
OBRERO MAS PRODUCTIVO:  XXXXXXXXXXXXXXXXXXXX



TOTAL GENERAL 99 OBREROS       99    99    99    99    99       99
OBRERO MAS PRODUCTIVO:  XXXXXXXXXXXXXXXXXXXX
```

Nota: Para efectos de este reporte, el archivo debe estar ordenado por planta, es decir, todos lo obreros de la planta 1 están juntos al principio del archivo, luego siguen los de la planta 2, y así sucesivamente.

Se pide el total de producción por cada planta para cada uno de los días, asimismo el nombre del obrero más productivo (el que tuvo la mayor producción semanal) por planta, al final el obrero más productivo y el total general de todas las plantas por cada día.

```
                REPORTE DE OBREROS MAS PRODUCTIVOS
   PLANTA      NOMBRE OBRERO MAS PRODUCTIVO   PROD. SEMANAL
-------------------------------------------------------------

   XXXXXXXXX      XXXXXXXXXXXXXXXXXXXXXXXXX          999
   XXXXXXXXX      XXXXXXXXXXXXXXXXXXXXXXXXX          999
     --
     --
   XXXXXXXXX      XXXXXXXXXXXXXXXXXXXXXXXXX          999
```

```
TOTAL   99  PLANTAS                                          999

OBRERO MAS PRODUCTIVO: XXXXXXXXXXXXXXXXXXXXXXXXX
PRODUCCION DEL OBRERO MAS PRODUCTIVO:  999
```

Nota: El archivo debe estar ordenado por planta. En este reporte se imprimen los datos del obrero más productivo de cada planta, y al final el obrero más productivo de toda la compañía.

```
                    REPORTE DE PRODUCTIVIDAD
                                          PRODUCCION       NIVEL
NUM.  NOMBRE              PLANTA          SEMANAL      PRODUCTIVIDAD
-------------------------------------------------------------------

 99   XXXXXXXXXXXXXXXXX XXXXXXXXX         999          XXXXXXXXXXX
 99   XXXXXXXXXXXXXXXXX XXXXXXXXX         999          XXXXXXXXXXX
         --
         --
 99   XXXXXXXXXXXXXXXXX XXXXXXXXX         999          XXXXXXXXXXX

TOTAL PLANTA  XXXXXXXXXXX
              999    OBREROS               999

NIVEL PRODUCTIVIDAD 1: 999 OBREROS
NIVEL PRODUCTIVIDAD 2: 999 OBREROS
NIVEL PRODUCTIVIDAD 3: 999 OBREROS
NIVEL PRODUCTIVIDAD 4: 999 OBREROS
NIVEL PRODUCTIVIDAD 5: 999 OBREROS
    .
    .
    .
TOTAL PLANTA  XXXXXXXXXXX
              999    OBREROS               999

NIVEL PRODUCTIVIDAD 1: --- OBREROS
NIVEL PRODUCTIVIDAD 2: --- OBREROS
NIVEL PRODUCTIVIDAD 3: --- OBREROS
NIVEL PRODUCTIVIDAD 4: --- OBREROS
NIVEL PRODUCTIVIDAD 5: --- OBREROS


TOTAL GENERAL  999  OBREROS               999

NIVEL PRODUCTIVIDAD 1: --- OBREROS
NIVEL PRODUCTIVIDAD 2: --- OBREROS
NIVEL PRODUCTIVIDAD 3: --- OBREROS
NIVEL PRODUCTIVIDAD 4: --- OBREROS
```

NIVEL PRODUCTIVIDAD 5: --- OBREROS

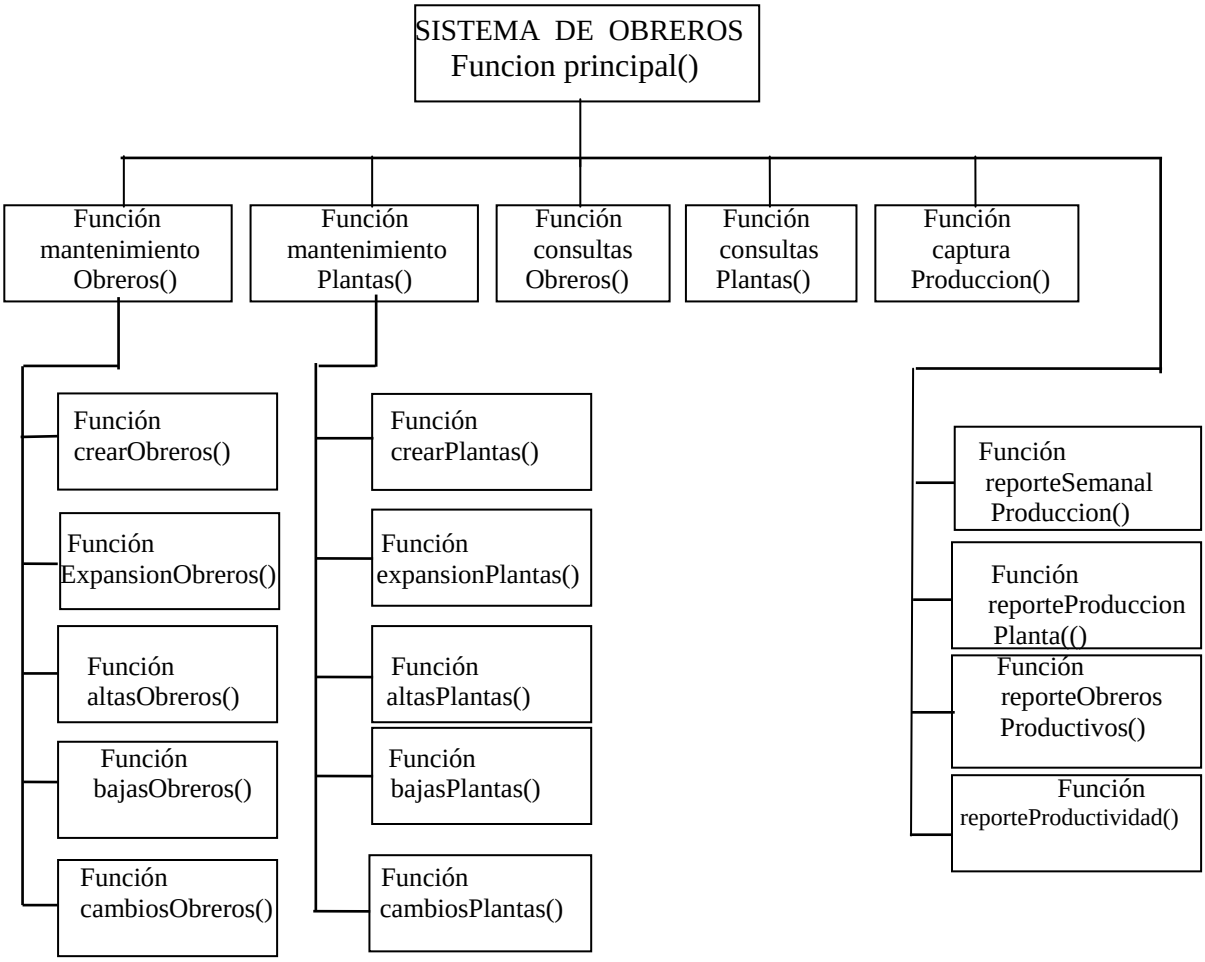
Nota: El archivo de obreros debe estar ordenado por planta.

Al emitir este roprte, primero se deben solicitar y leer los niveles de productividad en un arreglo de 5x2. Son 5 renglones, uno para cada nivel de productividad, desde el nivel 1 (renglón 1) hasta el nivel 5 (renglón 5). Cada nivel de productividad está constituido por un rango o intervalo de valores; en la columna 1 se coloca el límite inferior del rango y en la columna 2 se coloca el límite superior del rango.
Cada vez que se procesa un obrero, se calcula su producción semanal, enseguida se busca en el arreglo de niveles de productividad el nivel en el que la producción semanal es mayor o igual al límite inferior (columna 1) y menor o igual al límite superior (columna 2), en otras palabras, es decir, se busca en cuál rango cayó la producción semanal del obrero.
Como puede observarse en la estructura del reporte, se debe llevar un conteo de cuántos obreros estuvieron en cada uno de los niveles de productividad por planta y en general, es decir, por todas las plantas.

A continuación se presenta el algoritmo de la solución:

(Primero hágalo usted...después compare la solución)

Diagrama general del algoritmo sistema de obreros:

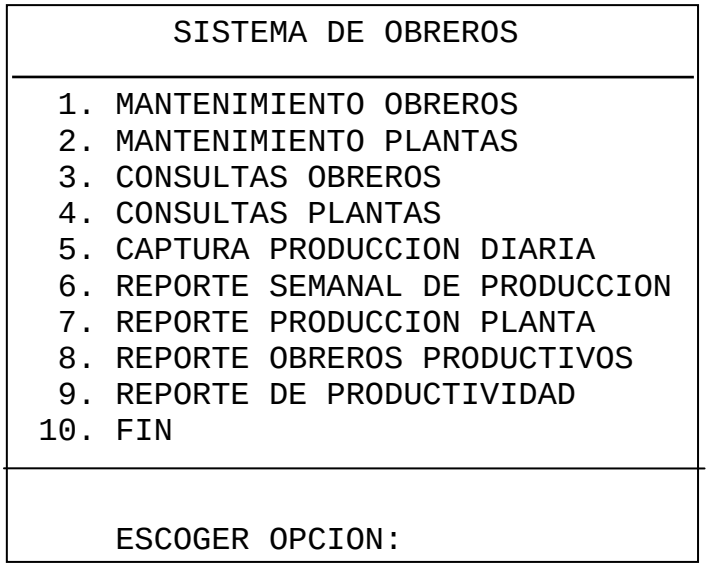


A continuación se presenta el diseño de la lógica de cada una de las funciones en pseudocódigo.

```
Algoritmo SISTEMA DE OBREROS
1. Declarar
     Tipos
       regObrero: Registro
                    numero: Entero
                    nombre: Cadena[30]
                    clavePlanta: Entero
                    produc: Arreglo[5] Entero
                  FinRegistro
       regPlanta: Registro
                    clavePlanta: Entero
                    nombrePlanta: Cadena[30]
                  FinRegistro
     Variables
       obreros: Archivo de regObrero
       plantas: Archivo de regPlanta
       i, num, numRegs, opcion, opcion2, opcion3: Entero
```

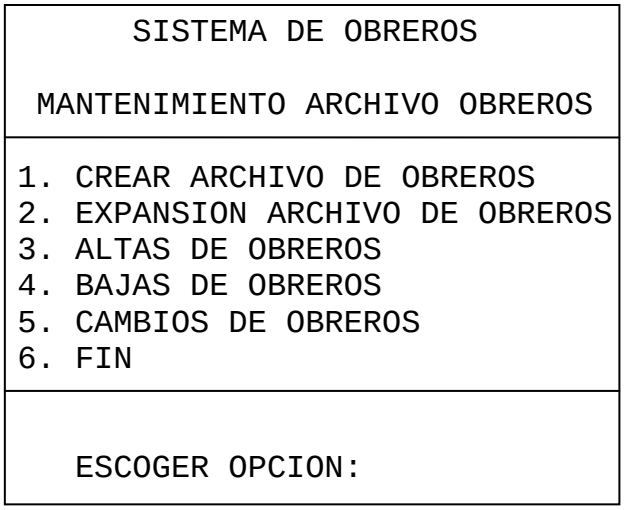
desea, seguro: Carácter

2. Función principal()
    a. do
        1. Imprimir MENU



        2. Leer opcion
        3. switch opcion
            1: mantenimObreros()
            2: mantenimPlantas()
            3: consultasObreros()
            4: consultasPlantas()
            5: capturaProduccion()
            6: reporteProduccion()
            7: reportePlanta()
            8: reporteProductivos()
            9: reporteProductividad()
        4. endswitch
    b. while opcion != 10
    c. Fin Función principal

3. Función mantenimObreros()
    a. do
        1. Imprimir MENU



2. Leer opcion2
3. switch opcion2
   1: crearObreros()
   2: expansionObreros()
   3: altasObreros()
   4: bajasObreros()
   5: cambiosObreros()
4. endswitch

b. while opcion2 != 6
c. Fin Función mantenimObreros

4. Función crearObreros()
   a. Crear (obreros, “AROBRE.DAT”, directo, w)
   b. Solicitar Número de registros que contendrá el archivo de obreros
   c. Leer numRegs
   d. regObrero.numero=0, regObrero.nombre = “”
      regObrero.clavePlanta = 0
   e. for i=1; i<=5; i++
      1. regObrero.produc[i] = 0
   f. endfor
   g. for i=1; i<=numRegs; i++
      1. Imprimir (obreros, regObrero)
   h. endfor
   i. Cerrar (obreros)
   j. Fin Función crearObreros

5. Función expansionObreros()
   a. Abrir (obreros, “AROBRE.DAT”, directo, rw)
   b. Solicitar Número de registros que se añadirán al archivo de obreros
   c. Leer numRegs

```
    d. regObrero.numero = 0, regObrero.nombre = ""
       regObrero.clavePlanta = 0
    e. for i=1; i<=5; i++
           1. regObrero.produc[i] = 0
    f. endfor
    g. Encontrar(obreros, TamañoAr(obreros))
    h. for i=1; i<=numRegs; i++
        1. Imprimir (obreros, regObrero)
    i. endfor
    j. Cerrar (obreros)
    k. Fin Función expansionObreros

6. Función altasObreros()
    a. Abrir (obreros, "AROBRE.DAT", directo, rw)
    b. do
        1. Imprimir PANTALLA de captura
```

SISTEMA DE OBREROS ALTAS ARCHIVO DE OBREROS
NUMERO: NOMBRE: CLAVE PLANTA:
¿OTRA ALTA(S/N)?:

```
        2. Leer num
        3. Encontrar(obreros, num)
        4. Leer (obreros, regObrero)
        5. if regObrero.numero == 0 then
              a. Leer regObrero.nombre, regObrero.clavePlanta
              b. regObrero.numero = num
              c. for i=1; i<=5; i++
                   1. regObrero.produc[i] = 0
              d. endfor
              e. Encontrar (obreros, num)
              f. Imprimir (obreros, regObrero)
        6. else
              a. Imprimir regObrero.nombre,
                          regObrero.clavePlanta
              b. Imprimir "YA ESTA DADO DE ALTA"
        7. endif
        8. Leer desea
    c. while desea == 'S'
```

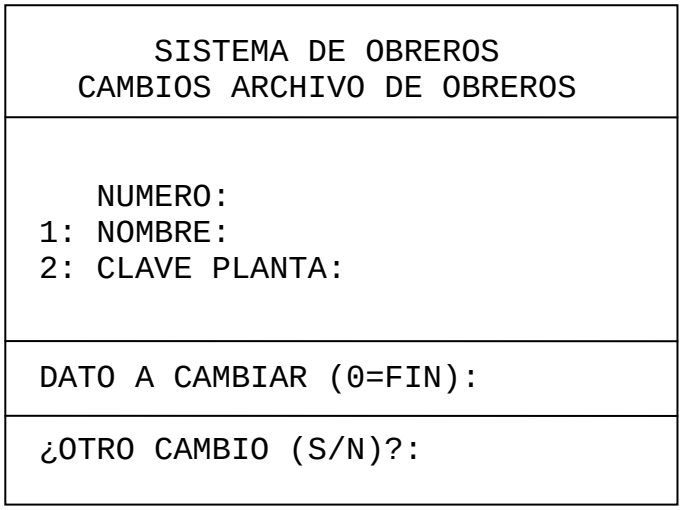
d. Cerrar (obreros)
e. Fin Función altasObreros

7. Función bajasObreros()
   a. Abrir (obreros, “AROBRE.DAT”, directo, rw)
   b. do
      1. Imprimir PANTALLA de captura

SISTEMA DE OBREROS BAJAS ARCHIVO DE OBREROS
NUMERO: NOMBRE: CLAVE PLANTA:
¿SON LOS DATOS, SEGURO(S/N)?:
¿OTRA BAJA (S/N)?:

      2. Leer num
      3. Encontrar (obreros, num)
      4. Leer (obreros, regObrero)
      5. if regObrero.numero == 0 then
         a. Imprimir “ESTE NUMERO NO EXISTE”
      6. else
         a. Imprimir regObrero.nombre,
                     regObrero.clavePlanta
         b. Leer seguro
         c. if seguro == ‘S’ then
            1. regObrero.numero = 0,
               regObrero.nombre = “”,
               regObrero.clavePlanta = 0
            2. for i=1; i<=5; i++
               1. regObrero.produc[i] = 0
            3. endfor
            4. Encontrar (obreros, num)
            5. Imprimir (obreros, regObrero)
         d. endif
      7. endif
      8. Leer desea
   c. while desea == ‘S’
   d. Cerrar (obreros)
   e. Fin Función bajasObreros

8. Función cambiosObreros()
    a. Abrir (obreros, “AROBRE.DAT”, directo, rw)
    b. do
        1. Imprimir PANTALLA de captura



        2. Leer num
        3. Encontrar (obreros, num)
        4. Leer (obreros, regObrero)
        5. if regObrero.numero == 0 then
            a. Imprimir “ESTE NUMERO NO EXISTE”
        6. else
            a. Imprimir regObrero.nombre,
                        regObrero.clavePlanta
            b. Leer opcion3
            c. while (opcion3 > 0) AND (opcion3 < 3)
                1. switch opcion3
                    1: Leer regObrero.nombre
                    2: Leer regObrero.clavePlanta
                2. endswitch
                3. Leer opcion3
            d. endwhile
            e. Encontrar (obreros, num)
            f. Imprimir (obreros, regObrero)
        7. endif
        8. Leer desea
    c. while desea == ‘S’
    d. Cerrar (obreros)
    e. Fin Función cambiosObreros

8. Función mantenimPlantas()
    a. do
        1. Imprimir MENU

SISTEMA DE OBREROS MANTENIMIENTO ARCHIVO PLANTAS
1. CREAR ARCHIVO 2. EXPANSION 3. ALTAS 4. BAJAS 5. CAMBIOS 6. FIN
ESCOGER OPCION:

        2. Leer opcion2
        3. switch opcion2
            1: crearPlantas()
            2: expansionPlantas()
            3: altasPlantas()
            4: bajasPlantas()
            5: cambiosPlantas()
        4. endswitch
    b. while opcion2 != 6
    c. Fin Función mantenimPlantas

10. Función crearPlantas()
    a. Crear (plantas, “ARPLAN.DAT”, directo, w)
    b. Solicitar Número de registros que contendrá el archivo
                de plantas
    c. Leer numRegs
    d. regPlanta.clavePlanta = 0,
       regPlanta.nombrePlanta = “”
    e. for i=1; i<=numRegs; i++
           1. Imprimir (plantas, regPlanta)
    f. endfor
    g. Cerrar (plantas)
    h. Fin Función crearPlantas

11. Función expansionPlantas()
     a. Abrir (plantas, “ARPLAN.DAT”, directo, rw)
     b. Solicitar Número de registros que se añadirán al
                 archivo de plantas
     c. Leer numRegs

d. regPlanta.clavePlanta = 0,
   regPlanta.nombrePlanta = “”
e. Encontrar(plantas, TamañoAr(plantas))
f. for i=1; i<=numRegs; i++
    1. Imprimir (plantas, regPlanta)
g. endfor
h. Cerrar (plantas)
i. Fin Función expansionPlantas

12. Función altasPlantas()
a. Abrir (plantas, “ARPLAN.DAT”, directo, rw)
b. do
    1. Imprimir PANTALLA de captura

SISTEMA DE OBREROS ALTAS ARCHIVO DE PLANTAS
CLAVE PLANTA: NOMBRE PLANTA:
¿OTRA ALTA (S/N)?:

    2. Leer num
    3. Encontrar(plantas, num)
    4. Leer (plantas, regPlanta)
    5. if regPlanta.clavePlanta == 0 then
        a. Leer regPlanta.nombrePlanta
        b. regPlanta.clavePlanta = num
        c. Encontrar (plantas, num)
        d. Imprimir (plantas, regPlanta)
    6. else
        a. Imprimir regPlanta.nombrePlanta
        b. Imprimir “YA ESTA DADA DE ALTA”
    7. endif
    8. Leer desea
c. while desea == ‘S’
d. Cerrar (plantas)
e. Fin Función altasPlantas

13. Función bajasPlantas()
    a. Abrir (plantas, “ARPLAN.DAT”, directo, rw)
    b. do
        1. Imprimir PANTALLA de captura

SISTEMA DE OBREROS BAJAS ARCHIVO DE PLANTAS
CLAVE PLANTA: NOMBRE PLANTA:
SON LOS DATOS, SEGURO(S/N)?:
¿OTRA BAJA (S/N)?:

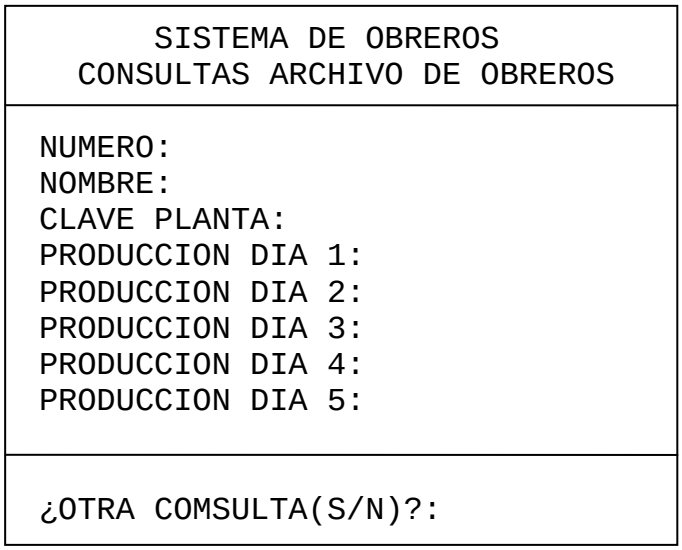
        2. Leer num
        3. Encontrar (plantas, num)
        4. Leer (plantas, regPlanta)
        5. if regPlanta.clavePlanta == 0 then
            a. Imprimir “ESTE NUMERO NO EXISTE”
        6. else
            a. Imprimir regPlanta.nombrePlanta
            b. Leer seguro
            c. if seguro == ‘S’ then
                1. regPlanta.clavePlanta = 0,
                   regPlanta.nombrePlanta = “”
                2. Encontrar (plantas, num)
                3. Imprimir (plantas, regPlanta)
            d. endif
        7. endif
        8. Leer desea
    c. while desea == ‘S’
    d. Cerrar (plantas)
    e. Fin Función bajasPlantas

14. Función cambiosPlantas()
    a. Abrir (plantas, “ARPLAN.DAT”, directo, rw)
    b. do
        1. Imprimir PANTALLA de captura

SISTEMA DE OBREROS CAMBIOS ARCHIVO DE PLANTAS
CLAVE PLANTA: 1: NOMBRE PLANTA:
DATO A CAMBIAR (1, 0=FIN):
¿OTRO CAMNIO(S/N)?:

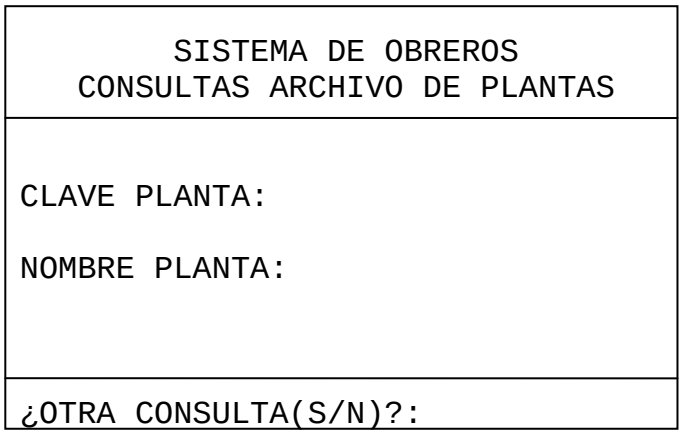
        2. Leer num
        3. Encontrar (plantas, num)
        4. Leer (plantas, regPlanta)
        5. if regPlanta.clavePlanta == 0 then
            a. Imprimir “ESTE NUMERO NO EXISTE”
        6. else
            a. Imprimir regPlanta.nombrePlanta
            b. Leer opcion3
            c. while (opcion3 > 0) AND (opcion3 < 2)
                1. switch opcion3
                    1: Leer regPlanta.nombrePlanta
                2. endswitch
                3. Leer opcion3
            d. endwhile
            e. Encontrar (plantas, num)
            f. Imprimir (plantas, regPlanta)
        7. endif
        8. Leer desea
    c. while desea == ‘S’
    d. Cerrar (plantas)
    e. Fin Función cambiosPlantas

15. Función consultasObreros()
    a. Abrir (obreros, “ARPLAN.DAT”, directo, rw)
    b. do
        1. Imprimir PANTALLA de captura



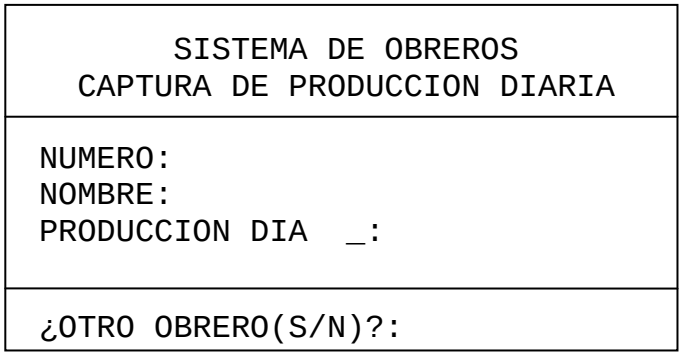
        2. Leer num
        3. Encontrar (obreros, num)
        4. Leer (obreros, regObrero)
        5. if regObrero.numero == 0 then
            a. Imprimir “ESTE NUMERO NO EXISTE”
        6. else
            a. Imprimir regObrero.nombre,
                        regObrero.clavePlanta
            b. for i=1; i<=5; i++
                    1. Imprimir regObrero.produc[i]
            c. endfor
        7. endif
        8. Leer desea
    c. while desea == ‘S’
    d. Cerrar (obreros)
    e. Fin Función consultasObreros

16. Función consultasPlantas()
    a. Abrir (plantas, “ARPLAN.DAT”, directo, rw)
    b. do
        1. Imprimir PANTALLA de captura



        2. Leer num
        3. Encontrar (plantas, num)
        4. Leer (plantas, regPlanta)
        5. if regPlanta.clavePlanta == 0 then
            a. Imprimir “ESTE NUMERO NO EXISTE”
        6. else
            a. Imprimir regPlanta.nombrePlanta
        7. endif
        8. Leer desea
    c. while desea == ‘S’
    d. Cerrar (plantas)
    e. Fin Función consultasPlantas

17. Función capturaProduccion()
    a. Abrir (obreros, “AROBRE.DAT”, directo, rw)
    b. Solicitar número (1-5) de día capturar
    c. Leer i
    d. do
        1. Imprimir PANTALLA de captura



        2. Leer num
        3. Encontrar (obreros, num)

```
        4. Leer (obreros, regObrero)
        5. if regObrero.numero == 0 then
             a. Imprimir “ESTE NUMERO NO EXISTE”
        6. else
             a. Imprimir regObrero.nombre
             b. Leer regObrero.produc[i]
             c. Encontrar (obreros, num)
             d. Imprimir(obreros, regObrero)
        7. endif
        8. Leer desea
     e. while desea == ‘S’
     f. Cerrar (obreros)
     g. Fin Función capturaProduccion

18. Función reporteProduccion()
     a. Declarar
          Variables
            totProDias: Arreglo[5] Entero
            totObr, prodSem, prodTot: Entero
     b. prodTot = 0 ; totObr = 0
     c. for i=1; i<=5; i++
            1. totProDias[i] = 0
     d. endfor
     e. Imprimir encabezado
     f. Abrir (obreros, “AROBRE.DAT”, directo, rw)
     g. Leer (obreros, regObrero)
     h. while NOT(Eof(obreros))
          1. Imprimir regObrero.numero, regObrero.nombre
          2. prodSem = 0
          3. for i=1; i<=5; i++
               a. Imprimir regObrero.produc[i]
               b. prodSem = prodSem + regObrero.produc[i]
               c. totProDias[i] = totProDias[i] +
                                  regObrero.produc[i]
          4. endfor
          5. Imprimir prodSem
          6. totObr = totObr + 1
             prodTot = prodTot + prodSem
          7. Leer (obreros, regObrero)
     i. endwhile
     j. Imprimir totObr
     k. for i=1; i<=5; i++
            a. Imprimir totProDias[i]
     l. endfor
     m. Imprimir prodTot
     n. Cerrar obreros
     o. Fin Función reporteProduccion
```

```
19. Función reportePlanta()
     a. Declarar
          Variables
            obreroMay, obreroMayPl: Cadena[30]
            toGeProDias, toPlProDias: Arreglo[5] Entero
            toObrGe, toObrPl, prodSem, prodSemPl, prodTot,
            plantaProceso, prodMay, prodMayPl: Entero
     b. prodTot = 0, toObrGe = 0, prodMay = 0
     c. for i=1; i<=5; i++
          1. toGeProDias[i] = 0
     d. endfor
     e. Imprimir encabezado
     f. Abrir (obreros, “AROBRE.DAT”, directo, rw)
        Abrir (plantas, “ARPLAN.DAT”, directo, rw)
     g. Leer (obreros, regObrero)
     h. while NOT(Eof(obreros))
          1. plantaProceso = regObrero.clavePlanta
          2. toObrPl = 0, prodSemPl = 0, prodMayPl = 0
          3. for i=1; i<=5; i++
                1. toPlProDias[i] = 0
          4. endfor
          5. while (plantaProceso==regObrero.clavePlanta)AND
                                          (NOT(Eof(obreros)))
               a. Imprimir regObrero.numero,
                           regObrero.nombre,
                           regObrero.clavePlanta
               b. prodSem = 0
               c. for i=1; i<=5; i++
                    1. Imprimir regObrero.produc[i]
                    2. prodSem = prodSem +
                                 regObrero.produc[i]
                    3. toPlProDias[i] = toPlProDias[i] +
                                        regObrero.produc[i]
               d. endfor
               e. Imprimir prodSem
               f. if prodSem > prodMayPl then
                     1. prodMayPl = prodSem
                     2. obreroMayPl = regObrero.nombre
               g. endif
               h. toObrPl = toObrPl + 1
                  prodSemPl = prodSemPl + prodSem
               i. Leer (obreros, regObrero)
          6. endwhile
          7. Encontrar(plantas, plantaProceso)
          8. Leer (plantas, regPlanta)
          9. Imprimir regPlanta.nombrePlanta, toObrPl
         10. for i=1; i<=5; i++
```

```
            a. Imprimir toPlProDias[i]
        11. endfor
        12. Imprimir prodSemPl
        13. Imprimir obreroMayPl
        14. if prodMayPl > prodMay then
               a. prodMay = prodMayPl
               b. obreroMay = obreroMayPl
        15. endif
        16. for i=1; i<=5; i++
              a. toGeProDias[i] = toGeProDias[i] +
                                  toPlProDias[i]
        17. endfor
        18. toObrGe = toObrGe + toObrPl
            prodTot = prodTot + prodSemPl
    i. endwhile
    j. Imprimir toObrGe
    k. for i=1; i<=5; i++
          a. Imprimir toGeProDias[i]
    l. endfor
    m. Imprimir prodTot
    n. Imprimir obreroMay
    o. Cerrar (obreros)
       Cerrar (plantas)
    p. Fin Función reportePlanta

20. Función reporteProductivos()
    a. Declarar
         Variables
           obreroMay, obreroMayPl: Cadena[30]
           totPlantas, prodSem, prodTot,
           plantaProceso, prodMay, prodMayPl: Entero
    b. prodTot = 0 ; prodMay = 0 ; totPlantas = 0
    c. Imprimir encabezado
    d. Abrir (obreros, “AROBRE.DAT”, directo, rw)
       Abrir (plantas, “ARPLAN.DAT”, directo, rw)
    e. Leer (obreros, regObrero)
    f. while NOT(Eof(obreros))
         1. plantaProceso = regObrero.clavePlanta
         2. prodMayPl = 0
         3. while (plantaProceso==regObrero.clavePlanta)AND
                                        (NOT(Eof(obreros)))
              a. prodSem = 0
              b. for i=1; i<=5; i++
                   1. prodSem = prodSem +
                                regObrero.produc[i]
              c. endfor
              d. if prodSem > prodMayPl then
```

```
                    1. prodMayPl = prodSem
                    2. obreroMayPl = regObrero.nombre
                e. endif
                f. Leer (obreros, regObrero)
            4. endwhile
            5. Encontrar(plantas, plantaProceso)
            6. Leer (plantas, regPlanta)
            7. Imprimir regPlanta.nombrePlanta, obreroMayPl,
                        prodMayPl
            8. if prodMayPl > prodMay then
                  a. prodMay = prodMayPl
                  b. obreroMay = obreroMayPl
            9. endif
           10. totPlantas = totPlantas + 1
               prodTot = prodTot + prodMayPl
      g. endwhile
      h. Imprimir totPlantas, prodTot, obreroMay, prodMay
      i. Cerrar (obreros)
         Cerrar (plantas)
      j. Fin Función reporteProductivos

21. Función reporteProductividad()
      a. Declarar
           Variables
             niveles: Arreglo[5,2] Entero
             totNivelPl, totNivelGe: Arreglo[5] Entero
             totObrGe, totObrPl, prodSem, prodSemPl, prodTot,
             plantaProceso, nivel: Entero
      b. prodTot = 0 ; totObrGe = 0
      c. for i=1; i<=5; i++
            1. Solicitar Límite inferior, Rango i
            2. Leer niveles[i,1]
            3. Solicitar Límite superior, Rango i
            4. Leer niveles[i,2]
      d. endfor
      e. for i=1; i<=5; i++
            1. totNivelGe[i] = 0
      f. endfor
      g. Imprimir encabezado
      h. Abrir (obreros, “AROBRE.DAT”, directo, rw)
         Abrir (plantas, “ARPLAN.DAT”, directo, rw)
      i. Leer (obreros, regObrero)
      j. while NOT(Eof(obreros))
          1. plantaProceso = regObrero.clavePlanta
          2. totObrPl = 0 ; prodSemPl = 0
          3. for i=1; i<=5; i++
               1. totNivelPl[i] = 0
```

```
        4. endfor
        5. while (plantaProceso==regObrero.clavePlanta)AND
                                        (NOT(Eof(obreros)))
             a. prodSem = 0
             b. for i=1; i<=5; i++
                  1. prodSem = prodSem + regObrero.produc[i]
             c. endfor
             d. i = 0
             e. do
                   1. i = i + 1
             f. while(prodSem >= niveles[i,1])AND
                                  (prodSem <= niveles[i,2])
             g. nivel = i
             h. Imprimir regObrero.numero, regObrero.nombre,
                         regObrero.clavePlanta, prodSem,
                         nivel
             i. totNivelPl[nivel] = totNivelPl[nivel] + 1
                totObrPl = totObrPl + 1
                prodSemPl = prodSemPl + prodSem
             j. Leer (obreros, regObrero)
        6. endwhile
        7. Encontrar(plantas, plantaProceso)
        8. Leer (plantas, regPlanta)
        9. Imprimir regPlanta.nombrePlanta, totObrPl,
                    prodSemPl
       10. for i=1; i<=5; i++
               a. Imprimir totNivelPl[i]
       11. endfor
       12. totObrGe = totObrGe + totObrPl
           prodTot = prodTot + prodSemPl
       13. for i=1; i<=5; i++
               a. totNivelGe[i] = totNivelGe[i] +
                                  totNivelPl[i]
       14. endfor
     k. endwhile
     l. Imprimir totObrGe, prodTot
     m. for i=1; i<=5; i++
          a. Imprimir totNivelGe[i]
     n. endfor
     o. Cerrar (obreros)
        Cerrar (plantas)
     p. Fin Función reporteProductividad
Fin
```

En la zona de descarga de la Web del libro, está disponible:
Programa en C: C806.C

Nota: Las funciones diseñadas en el algoritmo no incluyen algunas validaciones. Al codificar el programa, podrá utilizar algunas instrucciones adicionales que permitan y faciliten hacer algunas validaciones; por ejemplo, al abrir un archivo, se valida que éste exista, al imprimir un reporte detectar si la impresora no está lista (encendida, con papel y en línea), entre otras. La idea es que el programa permita detectar errores y recuperarse de los mismos; en el algoritmo se incluye la lógica principal del programa y aunque se contempla la detección de algunos errores, no se incluyen exhaustivamente todos los errores que pudieran ocurrir. Este programa podría mejorarse en las funciones que imprimen reportes en la impresora, controlando el salto de hoja e imprimiendo el encabezado en cada nueva hoja. El propósito de lo anterior es que el algoritmo no quede demasiado voluminoso.

Ejercicio 8.6.2

Elaborar un programa que maneje un sistema de control de ventas, que permita crear, hacer altas, bajas y cambios a un archivo de vendedores, otro de sucursales y otro de departamentos. El archivo de vendedores contiene los datos de todos los vendedores de la empresa y sus ventas, la empresa se divide en varias sucursales, y cada sucursal en varios departamentos, es por ello que cada vendedor tiene un dato que es la clave de la sucursal y otro que es la clave del departamento al que pertenece. El archivo de sucursales contiene los nombres de todas las sucursales que conforman la compañía, del mismo modo el archivo de departamentos contiene los nombres de los departamentos.

Vendedores	Sucursales	Departamentos
numero	claveSuc	claveDepto
nombreVend	sucursal	departamento
claveSuc		
claveDepto		
ventas[5]		

Donde:

numero	Es el número de identificación del vendedor.
nombreVend	Es el nombre del vendedor.
claveSuc	Es la clave de identificación de la sucursal.
claveDepto	Es la clave de identificación del departamento.
ventas[5]	Es un arreglo con 5 elementos: uno para almacenar la venta que el vendedor hizo en cada uno de los días de la semana.
.	
sucursal	Es el nombre de la sucursal del vendedor.
departamento	Es el nombre del departamento en el que labora.

La captura de la venta de los vendedores se debe hacer por día, es decir, se captura la venta del día 1 para todos los vendedores, luego el día 2 para todos los vendedores, y así sucesivamente hasta el día 5.

El algoritmo también debe proporcionar la posibilidad de hacer consultas de vendedores,
con teclear el número del mismo; además debe emitir los siguientes reportes:

```
                          REPORTE DE INCENTIVOS
NUMERO  NOMBRE                      SUCURSAL   VENTA SEMANA  INCENTIVO
-------------------------------------------------------------------------

  99    XXXXXXXXXXXXXXXXXXXX   XXXXXXXX      99999.99    99999.99
  99    XXXXXXXXXXXXXXXXXXXX   XXXXXXXX      99999.99    99999.99
          --
          --
TOTAL SUCURSAL    XXXXXXXXXXXXX
                     999  VENDEDORES          99999.99    99999.99
     .
     .
TOTAL SUCURSAL    XXXXXXXXXXXXX
                     999  VENDEDORES          99999.99    99999.99


TOTAL GENERAL   999  VENDEDORES               99999.99    99999.99
```

Nota:

El archivo debe estar ordenado por sucursal, es decir, todos los vendedores de la sucursal uno deberán estar juntos al inicio del archivo, luego estarán los de la sucursal dos, etcétera.

Se utilizará un arreglo de incentivos de 5x3. Al inicio se deberán leer los incentivos, un incentivo en cada renglón, es decir, cinco. Cada incentivo tiene tres columnas: en la columna uno se debe leer el límite inferior del rango del incentivo, en la columna dos el límite superior y en la columna tres el incentivo a pagar en dicho rango de ventas. Cada vez que se procesa un vendedor se busca en el arreglo de incentivos, el rango en el que cayó su venta semanal, para obtener el incentivo a pagarle de la columna tres.

```
                          REPORTE DE VENTAS
NUM.  NOMBRE              SUCURSAL   DEPTO.  VENTA SEMANA  NIVEL
VENTA
-------------------------------------------------------------------------

 99   XXXXXXXXXXXXXXXXX  XXXXXXXXX  XXXXXXX    99999.99
XXXXXXXXX
 99   XXXXXXXXXXXXXXXXX  XXXXXXXXX  XXXXXXX    99999.99
XXXXXXXXX
     .
```

```
      .
 99   XXXXXXXXXXXXXXXXX  XXXXXXXXX  XXXXXXX    99999.99
XXXXXXXXX

TOTAL DEPTO.   XXXXXXXXXXXXXX
                    999   VENDEDORES           99999.99
      .
      .
TOTAL DEPTO.   XXXXXXXXXXXXXX
                    999   VENDEDORES           99999.99
      .
      .
TOTAL SUCURSAL    XXXXXXXXXXXXXXXXX
                    999   VENDEDORES           99999.99
      .
      .
      .
TOTAL SUCURSAL    XXXXXXXXXXXXXXXXX
                    999   VENDEDORES           99999.99

TOTAL GENERAL       999   VENDEDORES           99999.99
```

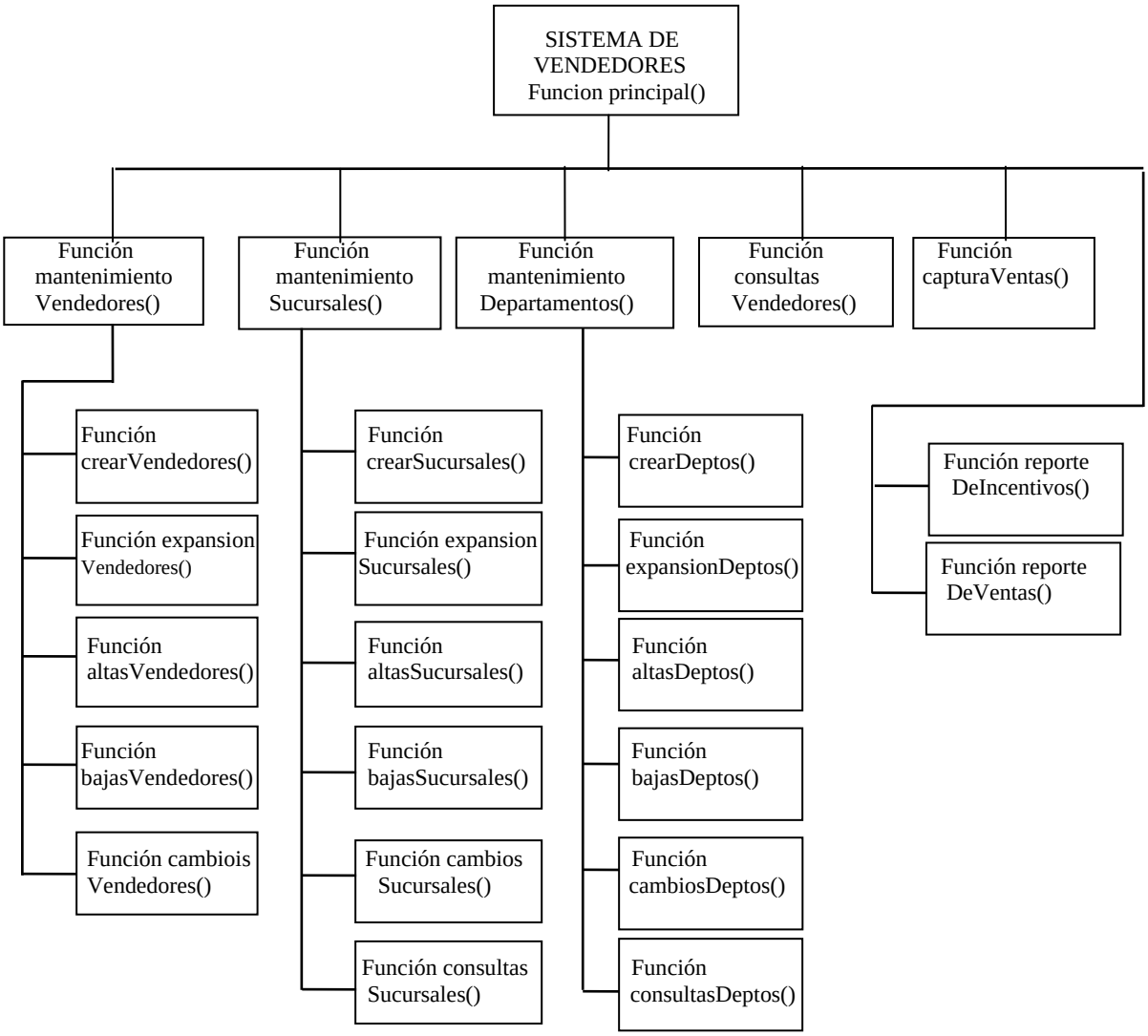
Nota:

El archivo debe estar ordenado por sucursal y por departamento, es decir, todos los vendedores de la sucursal uno deberán estar juntos al inicio del archivo, de la misma forma por departamento dentro de la sucursal, luego estarán los de la sucursal dos, etc.

NIVEL VENTA es un comentario de DEFICIENTE, BUENO o EXCELENTE de acuerdo al nivel de ventas alcanzado, por vendedor, por departamento y por sucursal. Al inicio de este reporte se deberá leer el nivel de venta esperado por vendedor, por departamento y por sucursal. Si la venta hecha está por debajo del nivel de venta esperado, el nivel de venta es DEFICIENTE; si es igual o mayor hasta un 20 % el nivel será BUENO, y si es mayor que éste punto, es EXCELENTE.
A continuación se presenta el algoritmo de la solución:

(Primero hágalo usted...después compare la solución)

Diagrama general del algoritmo sistema de vendedores:



A continuación se presenta el diseño de la lógica de cada una de las funciones en pseudocódigo.

```
Algoritmo SISTEMA DE VENDEDORES
1. Declarar
     Tipos
       regVend: Registro
                  numero    : Entero
                  nombre    : Cadena[30]
                  claveSuc  : Entero
                  claveDepto: Entero
                  ventas    : Arreglo[5] Real
                FinRegistro
       regSuc: Registro
                 claveSuc: Entero
                 Sucursal: Cadena[30]
```

```
            FinRegistro
      regDepto: Registro
                  claveDepto : Entero
                  departamento : Cadena[30]
                FinRegistro
    Variables
      vendedores: Archivo de regVend
      sucursales: Archivo de regSuc
      departamentos: Archivo de regDepto
      i, num, numRegs, opcion, opcion2, opcion3: Entero
      desea, seguro: Carácter
```

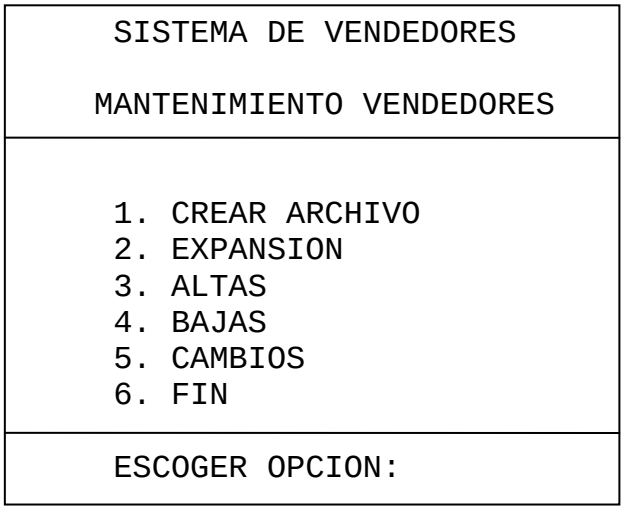
2. Función principal()
    a. do
        1. Imprimir MENU

SISTEMA DE VENDEDORES
1. MANTENIMIENTO VENDEDORES 2. MANTENIMIENTO SUCURSALES 3. MANTENIMIENTO DEPARTAMENTOS 4. CONSULTAS VENDEDORES 5. CAPTURA DE VENTAS 6. REPORTE DE INCENTIVOS 7. REPORTE DE VENTAS 8. FIN
ESCOGER OPCION:

        2. Leer opcion
        3. switch opcion
            1: mantenimVendedores()
            2: mantenimSucursales()
            3: mantenimDepartamentos()
            4: consultasVendedores()
            5: capturaventas()
            6: reporteIncentivos()
            7: reporteventas()
        4. endswitch
    b. while opcion != 8
    c. Fin Función principal

3. Función mantenimVendedores()
    a. do

1. Imprimir MENU



2. Leer opcion2
3. switch opcion2
    1: crearVendedores
    2: expansionVendedores
    3: altasVendedores
    4: bajasVendedores
    5: cambiosVendedores
4. endswitch

b. while opcion2 != 6
c. Fin Función mantenimVendedores

4. Función crearVendedores()
    a. Crear (vendedores, “ARVEND.DAT”, directo, w)
    b. Solicitar Número de registros que contendrá el archivo de vendedores
    c. Leer numRegs
    d. regVend.numero = 0, regVend.nombre = ””
       regVend.claveSuc = 0
    e. for i=1; i<=5; i++
        1. regVend.ventas[i] = 0
    f. endfor
    g. for i=1; i<=numRegs; i++
        1. Imprimir (vendedores, regVend)
    h. endfor
    i. Cerrar (vendedores)
    j. Fin Función crearVendedores

5. Función expansionVendedores()
    a. Abrir (vendedores, “ARVEND.DAT”, directo, rw)
    b. Solicitar Número de registros que se añadirán al archivo de vendedores

```
    c. Leer numRegs
    d. regVend.numero = 0, regVend.nombre = ""
       regVend.claveSuc = 0
    e. for i=1; i<=5; i++
           1. regVend.ventas[i] = 0
    f. endfor
    g. Encontrar(vendedores,TamañoAr(vendedores))
    h. for i=1; i<=numRegs; i++
           1. Imprimir (vendedores, regVend)
    i. endfor
    j. Cerrar (vendedores)
    k. Fin Función expansionVendedores

6. Función altasVendedores()
    a. Abrir (vendedores, "ARVEND.DAT", directo, rw)
    b. do
         1. Imprimir PANTALLA de captura
```

SISTEMA DE VENDEDORES ALTAS ARCHIVO DE VENDEDORES
NUMERO: NOMBRE: CLAVE SUCURSAL: CLAVE DEPARTAMENTO:
¿OTRA ALTA(S/N)?:

```
         2. Leer num
         3. Encontrar(vendedores, num)
         4. Leer (vendedores, regVend)
         5. if regVend.numero == 0 then
              a. Leer regVend.nombre, regVend.claveSuc,
                     regVend.claveDepto
              b. regVend.numero = num
              c. for i=1; i<=5; I++
                     1. regVend.ventas[i] = 0
              d. endfor
              e. Encontrar (vendedores, num)
              f. Imprimir (vendedores, regVend)
         6. else
              a. Imprimir regVend.nombreVend,
                          regVend.claveSuc,
                          regVend.claveDepto
              b. Imprimir "YA ESTA DADO DE ALTA"
```

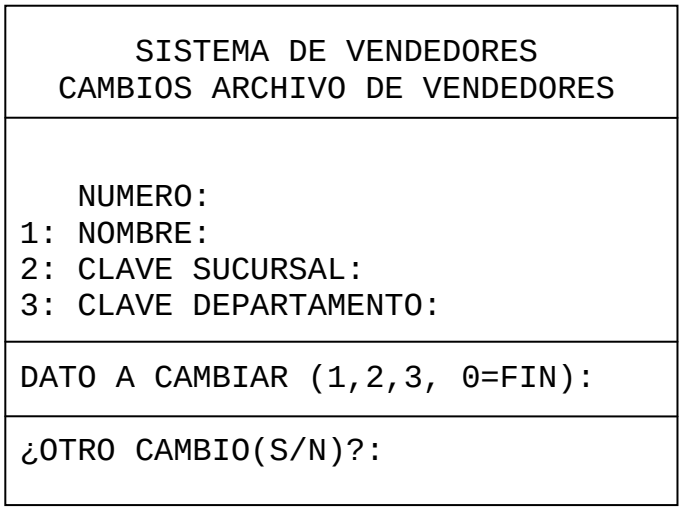
```
      7. endif
      8. Leer desea
   c. while desea == ‘S’
   d. Cerrar (vendedores)
   e. Fin Función altasVendedores

7. Función bajasVendedores()
   a. Abrir (vendedores, “ARVEND.DAT”, directo, rw)
   b. do
      1. Imprimir PANTALLA de captura
```

SISTEMA DE VENDEDORES BAJAS ARCHIVO DE VENDEDORES
NUMERO: NOMBRE: CLAVE SUCURSAL: CLAVE DEPARTAMENTO:
SON LOS DATOS, ¿SEGURO(S/N)?:
¿OTRA BAJA (S/N)?:

```
      2. Leer num
      3. Encontrar (vendedores, num)
      4. Leer (vendedores, regVend)
      5. if regVend.numero == 0 then
            a. Imprimir “ESTE NUMERO NO EXISTE”
      6. else
            a. Imprimir regVend.nombre, regVend.claveSuc
            b. Leer seguro
            c. if seguro == ‘S’ then
                  1. regVend.numero = 0,
                     regVend.nombre = ””,
                     regVend.claveSuc = 0,
                     regVend.claveDepto = 0
                  2. for i=1; I<=5; I++
                        a. regVend.ventas[i] = 0
                  3. endfor
                  4. Encontrar (vendedores, num)
                  5. Imprimir (vendedores, regVend)
            d. endif
      7. endif
      8. Leer desea
   c. while desea == ‘S’
```

d. Cerrar (vendedores)
e. Fin Función bajasVendedores

8. Función cambiosVendedores()
a. Abrir (vendedores, “ARVEND.DAT”, directo, rw)
b. do
1. Imprimir PANTALLA de captura



2. Leer num
3. Encontrar (vendedores, num)
4. Leer (vendedores, regVend)
5. if regVend.numero == 0 then
a. Imprimir “ESTE NUMERO NO EXISTE”
6. else
a. Imprimir regVend.nombre, regVend.claveSuc,
regVend.claveDepto
b. Leer opcion3
c. while (opcion3 > 0) AND (opcion3 < 4)
1. switch opcion3
1: Leer regVend.nombre
2: Leer regVend.claveSuc
3: Leer regVend.claveDepto
2. endswitch
3. Leer opcion3
d. endwhile
e. Encontrar (vendedores, num)
f. Imprimir (vendedores, regVend)
7. endif
8. Leer desea
c. while desea == ‘S’
d. Cerrar (vendedores)
e. Fin Función cambiosVendedores

9. Función mantenimSucursales()

a. do
    1. Imprimir MENU

SISTEMA DE VENDEDORES MANTENIMIENTO ARCHIVO SUCURSALES
1. CREAR ARCHIVO 2. EXPANSION 3. ALTAS 4. BAJAS 5. CAMBIOS 6. CONSULTAS 7. FIN
ESCOGER OPCION:

    2. Leer opcion2
    3. switch opcion2
        1: crearSucursales()
        2: expansionSucursales()
        3: altasSucursales()
        4: bajasSucursales()
        5: cambiosSucursales()
        6: consultasSucursales()
    4. endswitch
b. while Opcion2 != 7
c. Fin Función mantenimSucursales

10. Función crearSucursales()
    a. Crear (sucursales, “ARSUCU.DAT”, directo, rw)
    b. Solicitar Número de registros que contendrá el archivo
                de sucursales
    c. Leer numRegs
    d. regSuc.claveSuc=0, regSuc.sucursal = “”
    e. for i=1; i<=numRegs; i++
        1. Imprimir (sucursales, regSuc)
    f. endfor
    g. Cerrar (sucursales)
    h. Fin Función crearSucursales

11. Función expansionSucursales()
    a. Abrir (sucursales, “ARSUCU.DAT”, directo, rw)
    b. Solicitar Número de registros que se añadirán al
                archivo de sucursales
    c. Leer numRegs
    d. regSuc.claveSuc = 0, regSuc.sucursal = “”

e. Encontrar(sucursales, TamañoAr(sucursales))
f. for i=1; i<=numRegs; i++
    1. Imprimir (sucursales, regSuc)
g. endfor
h. Cerrar (sucursales)
i. Fin Función expansionSucursales

12. Función altasSucursales()
a. Abrir (sucursales, “ARSUCU.DAT”, directo, rw)
b. do
    1. Imprimir PANTALLA de captura

SISTEMA DE VENDEDORES ALTAS ARCHIVO DE SUCURSALES
CLAVE SUCURSAL: NOMBRE SUCURSAL:
¿OTRA ALTA(S/N)?:

    2. Leer num
    3. Encontrar(sucursales, num)
    4. Leer (sucursales, regSuc)
    5. if regSuc. claveSuc == 0 then
        a. Leer regSuc.Sucursal
        b. regSuc.claveSuc = num
        c. Encontrar (sucursales, num)
        d. Imprimir (sucursales, regSuc)
    6. else
        a. Imprimir regSuc.SUCURSAL
        b. Imprimir “YA ESTA DADA DE ALTA”
    7. endif
    8. Leer desea
c. while desea == ‘S’
d. Cerrar (sucursales)
e. Fin Función altasSucursales

13. Función bajasSucursales(()
a. Abrir (sucursales, “ARSUCU.DAT”, directo, rw)
b. do

1. Imprimir PANTALLA de captura

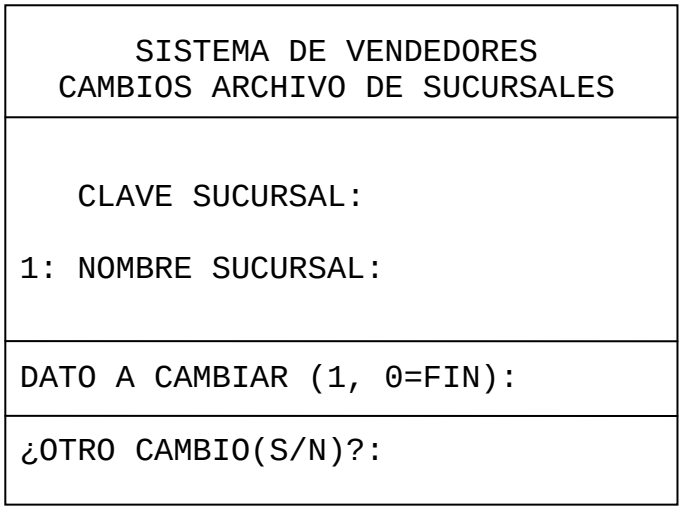
SISTEMA DE VENDEDORES BAJAS ARCHIVO DE SUCURSALES
CLAVE SUCURSAL: NOMBRE SUCURSAL:
SON LOS DATOS, ¿SEGURO(S/N)?:
¿OTRA BAJA (S/N)?:

2. Leer num
3. Encontrar (sucursales, num)
4. Leer (sucursales, regSuc)
5. if regSuc.claveSuc == 0 then
    a. Imprimir “ESTE NUMERO NO EXISTE”
6. else
    a. Imprimir regSuc.sucursal
    b. Leer seguro
    c. if seguro =='S' then
        1. regSuc.claveSuc=0, regSuc.Sucursal=””
        2. Encontrar (sucursales, num)
        3. Imprimir (sucursales, regSuc)
    d. endif
7. endif
8. Leer desea

c. while desea == ‘S’
d. Cerrar (sucursales)
e. Fin Función bajasSucursales

14. Función cambiosSucursales()
    a. Abrir (sucursales, “ARSUCU.DAT”, directo, rw)
    b. do

1. Imprimir PANTALLA de captura

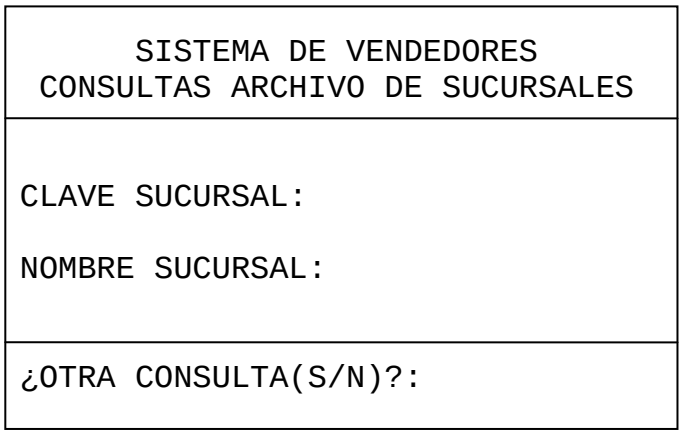


2. Leer num
3. Encontrar (sucursales, num)
4. Leer (sucursales,REGSUC)
5. if regSuc.claveSuc == 0 then
    a. Imprimir “ESTE NUMERO NO EXISTE”
6. else
    a. Imprimir regSuc.Sucursal,
    b. Leer opcion3
    c. while (opcion3 > 0) AND (opcion3 < 2)
        1. switch opcion3
            1: Leer regSuc.Sucursal
        2. endswitch
        3. Leer opcion3
    d. endwhile
    e. Encontrar (sucursales, num)
    f. Imprimir (sucursales, regSuc)
7. endif
8. Leer desea

c. while desea == ‘S’
d. Cerrar (sucursales)
e. Fin Función cambiosSucursales

15. Función consultasSucursales()
    a. Abrir (sucursales, “ARSUCU.DAT”, directo, rw)
    b. do

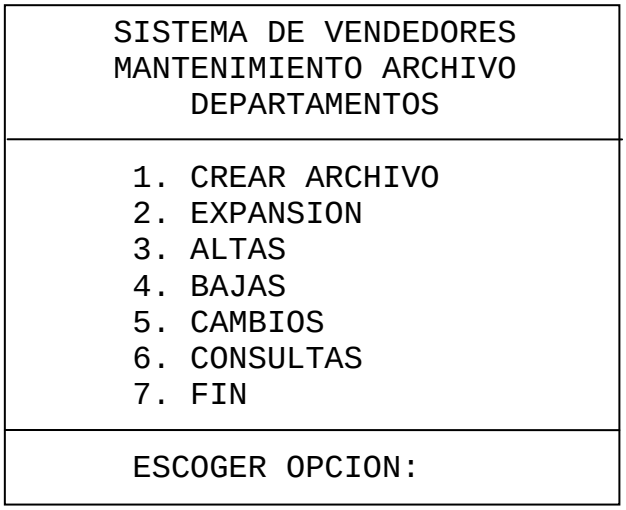
1. Imprimir PANTALLA de captura



2. Leer num
3. Encontrar (sucursales, num)
4. Leer (sucursales, regSuc)
5. if regSuc.claveSuc == 0 then
   a. Imprimir “ESTE NUMERO NO EXISTE”
6. else
   a. Imprimir regSuc.Sucursal
7. endif
8. Leer desea

c. while desea == ‘S’
d. Cerrar (sucursales)
e. Fin Función consultasSucursales

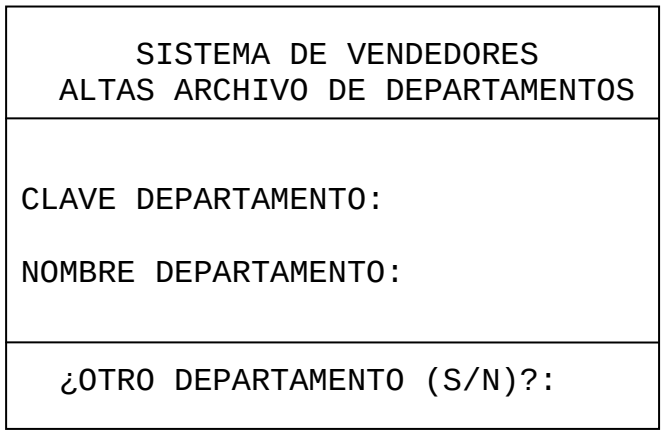
16. Función mantenimDepartamentos()
a. do
   1. Imprimir MENU



   2. Leer opcion2
   3. switch opcion2
      1: crearDepartamentos()

```
            2: expansionDepartamentos()
            3: altasDepartamentos()
            4: bajasDepartamentos()
            5: cambiosDepartamentos()
            6: consultasDepartamentos()
        4. endswitch
    b. while Opcion2 != 7
    c. Fin Función mantenimDepartamentos

17. Función crearDepartamentos()
    a. Crear (departamentos, “ARDEPA.DAT”, directo, w)
    b. Solicitar Número de registros que contendrá el
               archivo de departamentos
    c. Leer numRegs
    d. regDepto.claveDepto = 0, regDepto.departamento = “”
    e. for i=1; i<=numRegs; i++
           1. Imprimir (departamentos, regDepto)
    f. endfor
    g. Cerrar (departamentos)
    h. Fin Función crearDepartamentos

18. Función expansionDepartamentos()
    a. Abrir (departamentos, “ARDEPA.DAT”, directo, rw)
    b. Solicitar Número de registros que se añadirán al
               archivo de departamentos
    c. Leer numRegs
    d. regDepto.claveDepto = 0, regDepto.departamento = “”
    e. Encontrar(departamentos, TamañoAr(departamentos))
    f. for i=1; i<=numRegs; I++
           1. Imprimir (departamentos, regDepto)
    g. endfor
    h. Cerrar (departamentos)
    i. Fin Función expansionDepartamentos

19. Función altasDepartamentos()
    a. Abrir (departamentos, “ARDEPA.DAT”, directo, rw)
    b. do
```

1. Imprimir PANTALLA de captura



2. Leer num
3. Encontrar(departamentos, num)
4. Leer (departamentos, regDepto)
5. if regDepto.claveDepto == 0 then
    a. Leer regDepto.departamento
    b. regDepto.claveDepto = num
    c. Encontrar (departamentos, num)
    d. Imprimir (departamentos, regDepto)
6. else
    a. Imprimir regDepto. departamento
    b. Imprimir “YA ESTA DADA DE ALTA”
7. endif
8. Leer desea

c. while desea == ‘S’
d. Cerrar (departamentos)
e. Fin Función altasDepartamentos

20. Función bajasDepartamentos()
    a. Abrir (departamentos, “ARDEPA.DAT”, directo, rw)
    b. do

1. Imprimir PANTALLA de captura

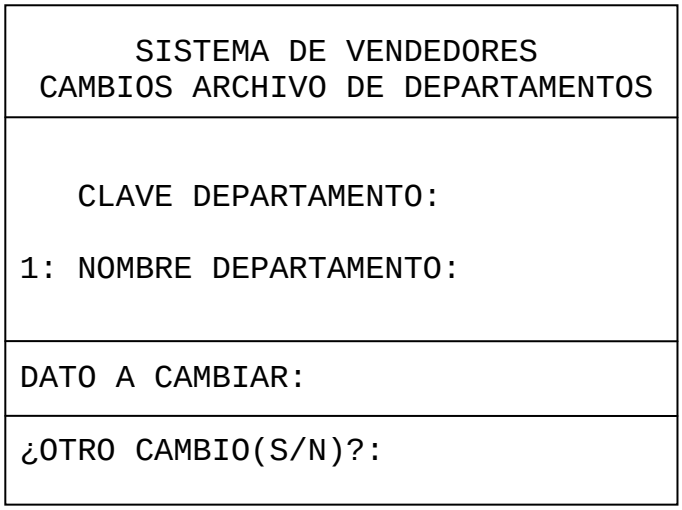
SISTEMA DE VENDEDORES BAJAS ARCHIVO DE DEPARTAMENTOS
CLAVE DEPARTAMENTO: NOMBRE DEPARTAMENTO:
SON LOS DATOS, ¿SEGURO(S/N)?:
¿OTRA BAJA (S/N)?:

2. Leer num
3. Encontrar (departamentos, num)
4. Leer (departamentos, regDepto)
5. if regDepto.claveDepto == 0 then
   a. Imprimir “ESTE NUMERO NO EXISTE”
6. else
   a. Imprimir regDepto. departamento
   b. Leer seguro
   c. if seguro == ‘S’ then
      1. regDepto.claveDepto = 0,
         regDepto.departamento = “”
      2. Encontrar (departamentos, num)
      3. Imprimir (departamentos, regDepto)
   d. endif
7. endif
8. Leer desea

c. while desea == ‘S’
d. Cerrar (departamentos)
e. Fin Función bajasDepartamentos

21. Función cambiosDepartamentos()
    a. Abrir (departamentos, “ARDEPA.DAT”, directo, rw)
    b. do

1. Imprimir PANTALLA de captura



2. Leer num
3. Encontrar (departamentos, num)
4. Leer (departamentos, regDepto)
5. if regDepto.claveDepto == 0 then
    a. Imprimir “ESTE NUMERO NO EXISTE”
6. else
    a. Imprimir regDepto.departamento
    b. Leer opcion3
    c. while (opcion3 > 0) AND (opcion3 < 2)
        1. switch opcion3
            1: Leer regDepto.departamento
        2. endswitch
        3. Leer opcion3
    d. endwhile
    e. Encontrar (departamentos, num)
    f. Imprimir (departamentos, regDepto)
7. endif
8. Leer desea

c. while desea == ‘S’
d. Cerrar (departamentos)
e. Fin Función cambiosDepartamentos

22. Función consultasDepartamentos()
    a. Abrir (departamentos, “ARDEPA.DAT”, directo, rw)
    b. do

1. Imprimir PANTALLA de captura

SISTEMA DE OBREROS CONSULTAS ARCHIVO DE DEPTOS.
CLAVE DEPTO.: NOMBRE DEPTO.:
¿OTRA CONSULTA(S/N)?:

2. Leer num
3. Encontrar (departamentos, num)
4. Leer (departamentos, regDepto)
5. if regDepto.claveDepto == 0 then
   a. Imprimir “ESTE NUMERO NO EXISTE”
6. else
   a. Imprimir regDepto. departamento
7. endif
8. Leer desea

c. while desea == ‘S’
d. Cerrar (departamentos)
e. Fin Función consultasDepartamentos

23. Función consultasVendedores()
a. Abrir (vendedores, “ARDEPA.DAT”, directo, rw)
b. do
   1. Imprimir PANTALLA de captura

SISTEMA DE VENDEDORES CONSULTAS ARCHIVO DE VENDEDORES
NUMERO: NOMBRE: CLAVE SUCURSAL: CLAVE DEPARTAMENTO: VENTAS DIA 1: VENTAS DIA 2: VENTAS DIA 3: VENTAS DIA 4: VENTAS DIA 5:
¿OTRO VENDEDOR (S/N)?:

```
        2. Leer num
        3. Encontrar (vendedores, num)
        4. Leer (vendedores, regVend)
        5. if regVend.numero == 0 then
             a. Imprimir “ESTE NUMERO NO EXISTE”
        6. else
             a. Imprimir regVend.nombre, regVend.claveSuc,
                         regVend.claveDepto
             b. for i=1; i<=5; I++
                   1. Imprimir regVend.ventas[i]
             c. endfor
        7. endif
        8. Leer desea
     c. while desea == ‘S’
     d. Cerrar (vendedores)
     e. Fin Función consultasVendedores

24. Función capturaVentas()
     a. Abrir (vendedores, “ARVEND.DAT”, directo, rw)
     b. Solicitar número (1-5) de día capturar
     c. Leer i
     d. do
        1. Imprimir PANTALLA de captura
```

SISTEMA DE VENDEDORES CAPTURA DE VENTAS DEL DIA
NUMERO: NOMBRE: VENTAS DIA _:
¿OTRO VENDEDOR (S/N)?:

```
        2. Leer num
        3. Encontrar (vendedores, num)
        4. Leer (vendedores, regVend)
        5. if regVend.numero == 0 then
             a. Imprimir “ESTE NUMERO NO EXISTE”
        6. else
             a. Imprimir regVend.nombre
             b. Leer regVend.ventas[i]
             c. Encontrar (vendedores, num)
             d. Imprimir(vendedores, regVend)
        7. endif
        8. Leer desea
     e. while desea == ‘S’
     f. Cerrar (vendedores)
     g. Fin Función capturaVentas
```

```
25. Función reporteIncentivos()
     a. Declarar
          Variables
            sucursalProceso, totVenSuc, totVen: Entero
            vtaSem, vtaSuc, vtaTot,
            incentivo, incentSuc, incentTot: Real
            incentivos: Arreglo[5,3] Real
     b. for i=1; i<=5; I++
          1. Solicitar Límite inferior, incentivo i
          2. Leer incentivos[i,1]
          3. Solicitar Límite superior, incentivo i
          4. Leer incentivos[i,2]
          5. Solicitar  incentivo i
          6. Leer incentivos[i,3]
     c. endfor
     d. totVen = 0 ; vtaTot = 0 ; incentTot = 0
     e. Imprimir encabezado
     f. Abrir (vendedores, "ARVEND.DAT", directo, rw)
        Abrir (sucursales, "ARSUCU.DAT", directo, rw)
     g. Leer (vendedores, regVend)
     h. while NOT(Eof(vendedores))
          1. sucursalProceso = regVend.claveSuc
             totVenSuc = 0, vtaSuc = 0, incentSuc = 0
          2. while (sucursalProceso==regVend.claveSuc)AND
                                        (NOT(Eof(vendedores)))
               a. vtaSem = 0
               b. for i=1; i<=5; I++
                    1. vtaSem = vtaSem + regVend.ventas[i]
               c. endfor
               d. i = 0
               e. do
                     1. i = i + 1
               f.while(vtaSem >= incentivos[i,1])AND
                                 (vtaSem <= incentivos[i,2])
               g. incentivo = incentivos[i,3]
               h. Imprimir regVend.numero, regVend.nombre,
                           regVend.claveSuc, vtaSem,
                           incentivo
               i. totVenSuc = totVenSuc + 1
                  vtaSuc = vtaSuc + vtaSem
                  incentSuc = incentSuc + incentivo
               j. Leer (vendedores, regVend)
          3. endwhile
          4. Encontrar(sucursales, sucursalProceso)
          5. Leer(sucursales, regSuc)
          6. Imprimir regSuc.Sucursal, totVenSuc, vtaSuc,
```

```
                    incentSuc
          7. totVen = totVen + totVenSuc
             vtaTot = vtaTot + vtaSuc
             incentTot = incentTot + incentSuc
     i. endwhile
     j. Imprimir totVen, vtaTot, incentTot
     k. Cerrar (vendedores)
        Cerrar (sucursales)
     l. Fin Función reporteIncentivos

26. Función reporteVentas()
     a. Declarar
          Variables
            deptoProceso, sucursalProceso,
            totVenDepto, totVenSuc, totVen: Entero
            vtaSem, VtaDepto, vtaSuc, vtaTot,
            nivelVta, nivelVtaDepto, nivelVtaSuc : Real
            obs, obsSuc, obsDep: Cadena[9]
     b. Solicitar nivel de venta por vendedor, por
                departamento y por sucursal
     c. Leer nivelVta, nivelVtaDepto, nivelVtaSuc
     d. totVen = 0 ; vtaTot = 0
     e. Imprimir encabezado
     f. Abrir (vendedores, "ARVEND.DAT", directo, rw)
        Abrir (sucursales, "ARSUCU.DAT", directo, rw)
        Abrir (departamentos, "ARDEPA.DAT", directo, rw)
     g. Leer (vendedores, regVend)
     h. while NOT(Eof(vendedores))
         1. sucursalProceso = regVend.claveSuc
            totVenSuc = 0, vtaSuc = 0
         2. while (sucursalProceso==regVend.claveSuc)AND
                                      (NOT(Eof(vendedores)))
              a. deptoProceso = regVend.claveDepto
                 totVenDepto = 0,  VtaDepto = 0
              b. while deptoProceso == regVend.claveDepto
                   1. vtaSem = 0
                   2. for i=1; i<=5; I++
                        a. vtaSem = vtaSem +
                                    regVend.ventas[i]
                   3. endfor
                   4. if vtaSem < nivelVta then
                        a. obs = "DEFICIENTE"
                   5. else
                        a. if vtaSem < (nivelVta * 1.2) then
                             1. obs = "BUENO"
                        b. else
                             1. obs = "EXCELENTE"
```

```
                    c. endif
                6. endif
                7. Imprimir regVend.numero,
                           regVend.nombre,
                           regVend.claveSuc,
                           regVend.claveDepto,
                           vtaSem, obs
                8. totVenDepto = totVenDepto + 1
                   VtaDepto = VtaDepto + vtaSem
                9. Leer (vendedores, regVend)
           c. endwhile
           d. if VtaDepto < nivelVtaDepto then
                1. obsDep = “DEFICIENTE”
           e. else
                1. if VtaDepto < (nivelVtaDepto*1.2) then
                     a. obsDep = “BUENO”
                2. else
                     a. obsDep = “EXCELENTE”
                3. endif
           f. endif
           g. Encontrar(departamentos,deptoProceso)
           h. Leer(departamentos, regDepto)
           i. Imprimir regDepto.departamento, totVenDepto,
                       VtaDepto, obsDep
           j. totVenSuc = totVenSuc + totVenDepto
              vtaSuc = vtaSuc + VtaDepto
      3. endwhile
      4. if vtaSuc < nivelVtaSuc then
           a. obsSuc = “DEFICIENTE”
      5. else
           a. if vtaSuc < (nivelVtaSuc * 1.2) then
                1. obsSuc = “BUENO”
           b. else
                1. obsSuc = “EXCELENTE”
           c. endif
      6. endif
      7. Encontrar(sucursales, sucursalProceso)
      8. Leer(sucursales, regSuc)
      9. Imprimir regSuc. Sucursal, totVenSuc, vtaSuc,
                  obsSuc
     10. totVen = totVen + totVenSuc
         vtaTot = vtaTot + vtaSuc
i. endwhile
j. Imprimir totVen, vtaTot
k. Cerrar (vendedores)
   Cerrar (sucursales)
   Cerrar (departamentos)
```

```
     l. Fin Función reporteVentas
Fin
```

En la zona de descarga de la Web del libro, está disponible:
Programa en C: C807.C

Nota:

Se recomienda como ejercicio, que se le añadan todas las validaciones posibles para que el algoritmo detecte los errores y se recupere de los mismos. Se puede tomar como base las validaciones que se hicieron anteriormente.