| {Ejercicio 1.  Una coordinadora de zona de una empresa de venta por catálogo tiene a su cargo 7 vendedoras. En cada campaña, cada vendedora está obligada a realizar exactamente 50 ventas. De cada venta se desea registrar: código de producto, precio unitario (cada vendedora puede poner su propio precio), cantidad de unidades vendidas y fecha de la venta. Si bien la cantidad de productos ofrecidos en una campaña es conocido, la cantidad de productos distintos que se terminan vendiendo es desconocido. 1) Haga un módulo que cargue en una matriz todas las ventas de cada una de las vendedoras. En cada fila se deben almacenar las ventas de cada vendedora. Las ventas  (generadas al azar) inicialmente no tienen orden alguno. 2) Haga un módulo que reciba la matriz y la retorne ordenada. Para cada vendedora ordene todas sus ventas por código de producto.  3) Haga un módulo que reciba las ventas ordenadas y genere un resumen por cada producto distinto vendido. Por cada producto se desea saber: cantidad total de unidades vendidas y monto total acumulado.   4) Haga un módulo recursivo que reciba los resúmenes de ventas y devuelva el producto con mayor monto acumulado.  5) Escriba un programa que invoque a los cuatro módulos implementados y compruebe el correcto funcionamiento del mismo.  L }  program ejercicio1; uses genericlinkedlist; const vendedoras=7; productos=50; //---------------------------- DECLARACION -------------------------------------------- type fecha=record dia:integer; mes:integer; anio:integer; end; producto=record codprod:integer; pu:real; cantvend:integer; fechavta:fecha; end;  acum=record codprod:integer; cant:integer; montoacum:real; end;  matriz=array[1..productos,1..vendedoras] of producto; lista=specialize linkedlist <producto>; listanueva=specialize linkedlist <acum>; vector=array[1..7] of integer;   //------------------------------MODULOS-------------------------------------------------   {1) Haga un módulo que cargue en una matriz todas las ventas de cada una de las vendedoras. En cada fila se deben almacenar las ventas de cada vendedora. Las ventas (generadas al azar) inicialmente no tienen orden alguno.}  PROCEDURE leerproducto(var a:producto); begin a.codprod:=random(10)+1; a.pu:=random(200-100+1)/2+100; a.cantvend:=random(10)+1; a.fechavta.dia:=random(31)+1; a.fechavta.mes:=random(12)+1; a.fechavta.anio:=random(2024-2000+1)+2000; end; //------------------------------------------------------------------------------------ PROCEDURE cargomatriz(var m:matriz); var a:producto; i,j:integer; begin  for i:=1 to productos do   begin  for j:=1 to vendedoras do  begin  leerproducto(a);  m[i,j]:=a;  end;  end; end; //xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx {2) Haga un módulo que reciba la matriz y la retorne ordenada. Para cada vendedora ordene todas sus ventas por código de producto.} //xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx   PROCEDURE ordenarfila(var m:matriz;l:integer); var   i,j,p: integer;  temp: producto; begin for i := 1 to productos - 1 do   begin  p := i;  for j := (i + 1) to Productos do   begin  if m[j,l].codprod < m[p,l].codprod then   begin  p := j;  end;  end;  // Intercambio de v[i] con v[p] si es necesario  if p <> i then   begin  temp := m[i,l];  m[i,l] := m[p,l];  m[p,l] := temp;  end;  end;  end; //------------------------------------------------------------------------------------   PROCEDURE ordenamatriz(var m:matriz); var j:integer; begin for j:=1 to vendedoras do begin ordenarfila(m,j); end; end;  //------------------------------------------------------------------------------------  PROCEDURE imprimir(var m: matriz); var  i, j: integer; begin    for j := 1 to vendedoras do   begin  write(' V',j,' |');  end; writeln();  writeln(' --------------------------------------------------');   for i := 1 to productos do   begin  for j := 1 to vendedoras do   begin  // Impresión de los códigos de productos por cada vendedora  write(' ', m[i,j].codprod:4, ' |'); // Imprimir el código del producto, alineado  end;  writeln();  end; end;    //xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx  {3) Haga un módulo que reciba las ventas ordenadas y genere un resumen por cada producto distinto vendido. Por cada producto se desea saber: cantidad total de unidades vendidas y monto total acumulado.} //xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx // SE PASA DIRECTAMENTE LA MATRIZ AL MERGE COMO SI FUERAN VARIOS VECTORES    //INICIALIZO VECTOR procedure inicializovc(var vc:vector); var i:integer; begin for i:=1 to vendedoras do  begin vc[i]:=1; end; end;   PROCEDURE minimo (m:matriz;var min:producto;var vc:vector); var posmin:integer; i,j:integer; begin min.codprod:=9999; posmin:=-1;  for j:=1 to vendedoras do begin i:=vc[j]; if (i<=productos) then  begin  if (m[i,j].codprod<=min.codprod) then  begin  min:= m[i,j];  posmin:=j;  end;  end;  end;  if min.codprod<>9999 then begin  vc[posmin]:= vc[posmin]+1;  end; end;    procedure resumenxprod(m:matriz; var l2:listanueva);  var  min:producto; productoactual,a:acum; vc:vector;  begin  //Creo una lista nueva con el registro acumulador  l2:=listanueva.create();   //inicio el vector en 1 inicializovc(vc);  //llamo al minimo minimo(m,min,vc);   // Corta en 9999 while (min.codprod <>9999) do begin productoactual.codprod:=min.codprod; productoactual.cant:=0; productoactual.montoacum:=0;  while (min.codprod=productoactual.codprod) do  begin  productoactual.cant:=productoactual.cant+min.cantvend;  productoactual.montoacum:=productoactual.montoacum+(min.cantvend\*min.pu);  minimo(m,min,vc);   end;  l2.add(productoactual); end;  //------------------------------------------------------------------------- // IMPRIMO l2.reset(); writeln('Producto |','Cantidad vendida',' | ','Total generado en $',' | '); while not l2.eol do begin a:=l2.current();  writeln(a.codprod:8, ' | ',a.cant:15,' | ',a.montoacum:19:2,' | '); l2.next(); end; end;  //xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx {4) Haga un módulo recursivo que reciba los resúmenes de ventas y devuelva el producto con mayor monto acumulado.} //xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx  PROCEDURE maximo (l:listanueva;var codmax:integer; var max:real); begin if not l.eol() then begin  if l.current().montoacum>max then   begin  max:=l.current().montoacum;  codmax:=l.current.codprod;  end;  l.next(); maximo(l,codmax,max); end; end; //---------------------------PROGRAMA------------------------------------------------ {5) Escriba un programa que invoque a los cuatro módulos implementados y compruebe el correcto funcionamiento del mismo.}  var m:matriz; l2:listanueva; codmax:integer; max:real; begin  //punto 1 cargomatriz(m); imprimir(m);  //Punto 2 ordenamatriz(m); writeln(' '); writeln('Ordenada'); imprimir(m); writeln(' ');  //punto 3 writeln('------------------ RESUMEN POR PRODUCTO --------------------'); resumenxprod(m,l2); writeln('-------------------------------------------------------------');   codmax:=1; max:=-1; l2.reset(); maximo(l2,codmax,max); writeln('El producto con un mayor monto vendido lleva el codigo de producto ',codmax,' con un acumulado de ventas de $',max:2:0);   end. |
| --- |