PRÁCTICA 10
REPASO

NOTA: UTILIZANDO PROGRAMACIÓN PROCEDURAL, RESUELVA LOS SIGUIENTES PROBLEMAS:

1.- Una agencia de viajes procesa la información de los pedidos de alojamiento de sus clientes. Para ello dispone de una estructura en la que almacena los pedidos. De cada pedido se conoce: Apellido y Nombre del Cliente, nombre del lugar turístico elegido, cantidad de personas a alojarse y precio máximo que el cliente desea pagar por cada persona. Esta información no se encuentra ordenada por ningún criterio.
Además, se dispone de una estructura que almacena la información de los hoteles de cada lugar turístico, la cual permite realizar una búsqueda eficiente por nombre del lugar turístico. De cada lugar turístico se conoce: nombre, distancia en km. desde La Plata y una lista de hoteles, donde por cada hotel se conoce su nombre, precio por persona y capacidad disponible.
Se pide procesar la información de los pedidos de alojamiento de los clientes, y para cada uno de ellos informe si el mismo se puede satisfacer o no. En caso de poder satisfacerse, se debe informar el nombre del hotel que se le asigna y actualizar la capacidad disponible del mismo.

2.- Tad T_Caja;
Interface
Type exportado caja;
Procedure crearCaja (var c: caja; pesoInicial:real; frágil:boolean; codigo:integer)
{Crea la caja c con "pesoInicial" como peso y "frágil" como estado de la caja}
Function verPeso (c: caja): real;
{Retorna el peso de la caja c}
Function verEstado (c: caja): boolean;
{Retorna *true* si el estado de la caja es frágil y *false* en caso contrario}
Function verCodigo (c: caja): integer;
{Retorna el código de la caja c}
Procedure modificarPeso (var c: caja; peso: real)
{modifica el peso de la caja c con el nuevo valor recibido en "peso"}
Procedure modificarEstado (var c: caja; fragil: boolean)
{modifica el estado de la caja c con el nuevo valor recibido en "frágil"}
Procedure asignarCaja (var c1: caja; c2: caja)
{Devuelve la caja c1 a la que se le han cargado los valores de la caja c2}

Utilizando el TAD T_Caja (no hay que implementar) resuelva el siguiente problema. Una empresa empaquetadora dispone en su almacén de una serie de cajas almacenadas sin ningún orden en particular. Se pide:

a) Generar una nueva estructura de manera eficiente que almacene las cajas de la empresa según su peso, es decir, la nueva estructura debe contener para cada peso todas las cajas que tienen dicho peso.
b) Con la estructura generada en el punto a), informar cuántas cajas tienen estado frágil y su peso es 35 kilos. (Recuerde que puede no existir cajas con 35 kilos de peso).
Nota: definir todas las estructuras utilizadas. MODULARIZAR.

3.- Una agencia de automóviles posee una lista de autos para la venta. De cada auto se conoce la marca, modelo, velocidad máxima, año y tipo de combustible. Esta información está ordenada por marca y luego por año. Se solicita realizar un programa que:
a. Genere una estructura eficiente con la información de los autos ordenada por año, donde por cada año se tengan los autos ordenados por marca.
A partir de la estructuras de datos disponibles:
b. Informar la cantidad de autos a gas del año 1998.
c. Imprimir un reporte con la siguiente información:
Marca:
Año:
Modelo Velocidad Máx. Combustible
. .....
Total Año:
Año:
. .....
Total Año:
Total Marca:

4.- Una empresa de viajes de La Plata dispone de un catálogo de destinos turísticos; de cada destino se conoce: el nombre, distancia en km y precio por persona.
Además, se dispone de una lista de paquetes vendidos. De cada paquete vendido se conoce: nombre de destino, fecha de salida y cantidad de personas que viajan; esta información está ordenada por nombre de destino.
Se pide:
a) Generar una estructura eficiente, donde por cada distancia en km se disponga la colección de paquetes vendidos.
b) A partir de la estructura generada en a), informar la cantidad de paquetes vendidos para distancias entre 1000 y 2300 km (excluyendo 1000 y 2300).
c) Informar para cada destino el monto total recaudado, y cual destino es el más solicitado (con mayor cantidad de personas que viajan a ese destino).
Nota: todos los paquetes vendidos están en el catálogo de destinos turísticos de la empresa.

5.-Tad T_Pizza;
Type exportado pizza;
Procedure CrearPizza (var p:pizza; nom:string, cod: integer)
//Crea la pizza p con nombre “nom”, codigo cod y sin ningún producto para su elaboración.
Procedure AgregarProducto (var p:pizza; nom:string)
//Agrega a la pizza p el producto de nombre nom.
Function verCodigo (p:pizza): integer;
//Devuelve el codigo de la pizza p.
Procedure verNombre(p:pizza; var nom:string);
//Devuelve el nombre de la pizza p.
Function verCantidadProductos (p:pizza): integer;
//Devuelve la cantidad de productos necesarios para hacer la pizza p.
Procedure dameProducto (p:pizza; var prod:string; i:integer);
//De todos los productos necesarios para realizar la pizza p devuelve el nombre del producto ubicado en la posición i.
Procedure asignarPizza (var p1:pizza; p2:pizza);
//Asigna el contenido de la pizza p2 a la pizza p1.
Implemente el TAD T_Pizza y luego resuelva el siguiente problema: Una pizzería dispone de una carta con diversas variedades de pizzas.

Además, la pizzería dispone de una estructura eficiente con los productos necesarios para la elaboración de las pizzas, donde de cada producto se conoce: nombre y stock actual; esta información se encuentra ordenada por nombre.
Se pide procesar los pedidos de sus clientes. De cada pedido se debe leer: apellido del cliente y el código de la pizza. La lectura de los pedidos finaliza cuando se lee el apellido “Chichizola” el cual debe procesarse. A medida que se van atendiendo los pedidos, indicar si el pedido se llevó a cabo o no.
Aclaración: Una pizza se puede vender si la pizzería tiene disponible cada uno de los productos que la misma necesita (la cantidad que necesita de cada producto siempre es 1). En caso que el pedido se lleve a cabo, se debe actualizar el stock de productos utilizados para la elaboración descontando una unidad de cada producto utilizado.
Nota: defina todas las estructuras necesarias para resolver el problema. MODULARICE.