

**PROGRAMACIÓN I**

EVALUACIÓN DIAGNÓSTICA

# UNIDAD 1

**REPASO E INTRODUCCIÓN A LA MATERIA**

**Autor de contenidos:**

Nicolás Battaglia

# OBJETIVOS

Realizar un repaso general de programación estructurada

# ENUNCIADO

1. Describir con sus palabras, cada uno de los siguientes items
   * Programación
   * Algoritmos
   * Vectores,
   * Funciones,
   * Procedimientos
   * Variables
2. Realizar el código necesario para resolver el siguiente enunciado. Utilizar C.

Una fábrica de autos tiene los siguientes lotes de datos

**Lote 1** composición del auto

* + Modelo
  + Pieza
  + Cantidad a usar de esa pieza

**Lote 2** stock de modelos terminados

* + Modelo
  + Stock

**Lote 3** stock de piezas

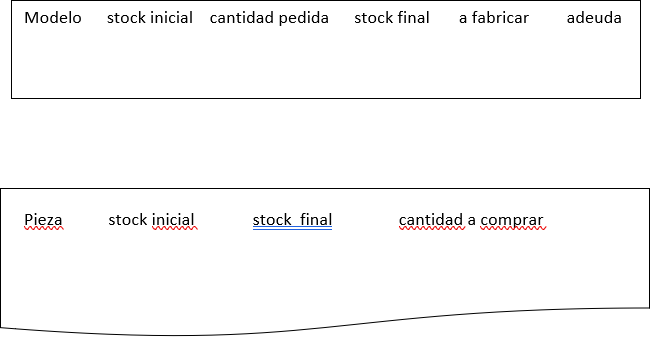
* + Pieza
  + Stock

**Lote 4** pedidos de las sucursales, finaliza con modelo = 0

* + Modelo
  + Cantidad pedida en total

La fábrica construye 20 modelos de auto, que utilizan entre sí 100 piezas distintas

**Se desea saber**



Desarrollo:

1. Describir con sus palabras, cada uno de los siguientes items
   * Programación

La programación es la actividad de codificar un programa en algún lenguaje para que realice alguna función, resuelva un problema o automatice una función. La programación puede realizarse en distintos lenguajes, de distintas complejidades y es muchas veces asociada con una parte artística, ya que no hay una sola forma o una forma correcta de programar.

* + Algoritmos

Los algoritmos son métodos o formas a través de las cuales los programadores, a través del código, resuelven un problema.

* + Vectores

Los vectores o arrays son listas cuyas posiciones están ordenadas y pueden guardar un valor, ya sea un entero o un carácter, que suelen comenzar en cero y son un elemento de uso habitual en la programación.

* + Funciones

Las funciones son porciones de código que pueden ser llamados por la función principal o main, que cumplen una función. Idealmente estas funciones son atómicas, cumplen una sola función en la resolución del problema (como por ejemplo pedir datos, imprimir resultados, entre otras) y permiten el trabajo modular y exponencial. Pueden tener valores de retorno o no (void).

* + Procedimientos

Los procedimientos son funciones que no tienen un valor de retorno a la función principal o main del programa que se está codificando.

* + Variables

Las variables son elementos de la programación que son capaces de alojar valores de distintos tipos y que pueden cambiar, por ejemplo, pueden alojar valores de tipo entero, de tipo flotante o carácter, y el valor puede ir cambiando, por ejemplo, la variable “stock=0” , al comenzar el programa tiene un valor nulo, y al finalizar la ejecución del programa puede terminar con un valor distinto, por ejemplo, “stock=8”.

1. Realizar el código necesario para resolver el siguiente enunciado. Utilizar C.

*El código se encuentra adjunto en la carpeta ZIP.*

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Una fÃ¡brica de autos tiene los siguientes lotes de datos

Lote 1 composiciÃ³n del auto

â€¢ Modelo

â€¢ Pieza

â€¢ Cantidad a usar de esa pieza

Lote 2 stock de modelos terminados

â€¢ Modelo

â€¢ Stock

Lote 3 stock de piezas

â€¢ Pieza

â€¢ Stock

Lote 4 pedidos de las sucursales, finaliza con modelo = 0

â€¢ Modelo

â€¢ Cantidad pedida en total

La fÃ¡brica construye 20 modelos de auto, que utilizan entre sÃ­ 100 piezas distintas

Se desea saber:

â€¢Modelo, Stock Inicial, Cantidad Pedida, Stock Final, a fabricar, a deuda

â€¢Pieza, stock inicial, stock final, cantidad a comprar

//Adeudo el uso de las funciones

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <stdio.h>

#include <conio.h>

#include <math.h>

#include <stdlib.h>

int main()

{

//Defino variables

//variables iniciles para hacer las matrices del ejercicio

int cantidadModelo=4;

int cantidadPieza=4;

int modelo=0;

int pieza=0;

int i=0;

int j=0;

//matriz componentes, que modelos de autos utilizan que componentes:

int MC[cantidadModelo][cantidadPieza];

//int piezasPorModelo=0;

//stock inicial y final de modelos de autos:

int VSIM[cantidadModelo];

int stockModelo;

int VSFM[cantidadModelo];

//stock inicial y final de piezas de autos:

int VSIP[cantidadPieza];

int stockPieza;

int VSFP[cantidadPieza];

//cantidad de modelos pedidos

int cantidadPedida=0;

//detalles de lista //detalles listado 1

int AFabricar=0;

int ADeuda=0;

//recorrido de piezas

int x=0;

int minimaCantidad=0;

//int cantidadAFabricar=0;

//posible cantidad de piezas a comprar

int VPCP[cantidadPieza];

//impresion final de datos

int PiezasAComprar[cantidadPieza];

//auxiliares

int CPAF=0;

int carga=0;

//inicio del programa

printf("\*\*\*Bienvenido al software de Fabrica de Autos\*\*\*\n");

/\*

printf("\*\*\*Ingreso de cantidad de MODELOS y PIEZAS a trabajar\*\*\*\n");

while(carga!=1){

printf("Por favor ingrese la cantidad de MODELOS con la que desea trabajar (se recomienda 4):\n");

scanf("%i",&cantidadModelo);

printf("Por favor ingrese la cantidad de PIEZAS con la que desea trabajar (se recomienda 4):\n");

scanf("%i",&cantidadPieza);

printf("Las cantidades seleccionadas son %i de modelos y %i de piezas", cantidadModelo,cantidadPieza);

printf("\nSi la carga fue correcta ingrese el valor 1, si fue incorrecta ingrese el valor 0...\n");

scanf("%i", &carga);

}\*/

//Lote 1 (de todo:modelo, piezas, stock)--> (i=1,20)(j=1,100) Matriz composición --> MC (Modelo, Parte) = Cantidad a Utilizar

printf("\*\*\*Ingreso de piezas por modelo\*\*\*\n");

while(carga!=2){

for(i=0;i<cantidadModelo;i++){

for (j=0; j<cantidadPieza;j++){

printf("Cantidad de veces que la pieza %i es utilizada en el modelo numero %i :", j+1, i+1);

scanf("%i", &MC[i][j]);

}

printf("\n");

}

//Impresión de la matriz resultante

for(i=0;i<cantidadModelo;i++){

printf("Composicion del modelo n%i :", i+1);

for (j=0; j<cantidadPieza;j++){

printf("\* %i \*", MC[i][j]);

}

printf("\n");

}

printf("\nSi la carga fue correcta ingrese el valor 2, si fue incorrecta ingrese el valor 0...\n");

scanf("%i", &carga);

}

//Lote 2 (modelo)--> (i=1,20) Vector Stock Inicial de Modelo --> VSIM (Stock) = Cantidad inicial de Stock de Modelos

//Ingreso de la cantidad de modelos en stock

printf("\*\*\*Ingreso de stock por modelo\*\*\*\n");

while(carga!=3){

for(i=0;i<cantidadModelo;i++){

printf("Stock del modelo n%i: ", i+1);

scanf("%i", &VSIM[i]);

//Lote 2 (modelo)--> (i=1,20) Vector Stock Final de Modelo --> VSFM (Stock) = Cantidad final de Stock de Modelos

VSFM[i]=VSIM[i];

}

//Impresion para corroboracion

printf("El stock por modelo es: \n");

for(i=0;i<cantidadModelo;i++){

printf("Stock del modelo n%i es de: %i\n", i+1, VSIM[i]);

}

printf("\n Si la carga fue correcta ingrese el valor 3, si fue incorrecta ingrese el valor 0...\n");

scanf("%i", &carga);

}

//Lote 3 (piezas)--> (i=1,100) Vector Stock Inicial de Piezas --> VSIP (Stock) = Cantidad inicial de Stock de Piezas

//Ingreso de la cantidad de piezas en stock

printf("\*\*\*Ingreso de stock por pieza\*\*\*\n");

while(carga!=4){

for(i=0;i<cantidadPieza;i++){

printf("\n Stock de la pieza n%i: ", i+1);

scanf("%i", &VSIP[i]);

//Lote 3 (piezas)--> (i=1,100) Vector Stock Final de Piezas --> VSFP (Stock) = Cantidad final de Stock de Piezas

VSFP[i]=VSIP[i];

}

//Impresion para corroboracion

{

printf("El stock por pieza es: \n");

for(i=0;i<cantidadPieza;i++){

printf("\n Stock de la pieza n%i es de: %i ", i+1, VSIP[i]);

}

printf("\n Si la carga fue correcta ingrese el valor 4, si fue incorrecta ingrese el valor 0...\n");

scanf("%i", &carga);

}

//Lote 4 (vendido)--> Ciclo while, modelo!=0, ingreso el modelo y la cantidad pedida del modelo -->

//Proceso

printf("\*\*\*Pedido de autos\*\*\*\n");

printf("Por favor ingrese el numero del modelo de auto que desea comprar. Para salir, ingrese 0 \n");

scanf("%i",&modelo);

while(modelo!=0){

printf("Por favor ingrese la cantidad autos que desea comprar:\n");

scanf("%i",&cantidadPedida);

//Corroborar stock

//Si la cantidad pedida es menor o igual que el vector final de modelos, resto la cantidad pedida al vector final y prosigo con el programa

//AFabricar=0;Adeuda=0;VSFM(modelo)=VSFM(modelo)-CantidadPedida

if(cantidadPedida<=VSFM[modelo]){

VSFM[modelo]=VSFM[modelo]-cantidadPedida;

AFabricar=0;

ADeuda=0;

//Impresion del primer listado

//Modelo, VSIM, CantidadPedida, VSFM, AFabricar, Adeuda

printf("\*\*\*Impresion de listado 1: cantidad de autos\*\*\*\n");

printf("Del modelo numero %i: \n se pidieron %i cantidad de autos \n de los cuales habia un stock inicial de % \n dejando un stock de %i \n se deben fabricar %i \n se adeudan %i\n", modelo, cantidadPedida, VSIM[modelo], VSFM[modelo], AFabricar, ADeuda);

}else{

//Sino

//Cantidad de Posible de Autos a Fabricar CPAF = CantidadPedida-VSFM(modelo)

//VSFM(modelo)=0

CPAF= cantidadPedida-VSFM[modelo];

VSFM[modelo]=0;

minimaCantidad=CPAF;

//Corroborar cantidad minima de piezas

//Primero corroboro utilización de la pieza

//Si MatrizComponentes MC(modelo,pieza)=!0

for(i=0;i<cantidadPieza;i++){

if(MC[modelo][i]!=0){

//determino la cantidad de piezas a utilizar

x=VSFP[i]/MC[modelo][i];

//x tiene que ser la menor posible, porque son autos enteros;

if(x<minimaCantidad){

minimaCantidad=x;

}

}

}

//AFabricar=minimaCantidad

AFabricar=minimaCantidad;

//Adeuda= CantidadPedida-AFabricar

ADeuda=CPAF-minimaCantidad;

//Finaliza el proceso de control de piezas, es hora de descontar las piezas

for(i=0;i<cantidadPieza;i++){

//Verifico si se utiliza

if(MC[modelo][i]!=0){

//Si se utiliza, resto:

VSFP[i]=VSFP[i]-(AFabricar\*MC[modelo][i]);

//A la vez que descuento puedo ir en un vector auxiliar anotando las piezas que puede que tenga que comprar

//Vector de Posibles Compras de Piezas VPCP(pieza)=VPCP(pieza)+(Adeuda\* MC(modelo,pieza))

VPCP[i]=VPCP[i]-(ADeuda\*MC[modelo][i]);

}

}

printf("\*\*\*Impresion de listado 1: cantidad de autos\*\*\*\n");

printf("Del modelo numero %i: \n se pidieron %i cantidad de autos \n de los cuales habia un stock inicial de % \n dejando un stock de %i \n se deben fabricar %i \n se adeudan %i\n", modelo, cantidadPedida, VSIM[modelo], VSFM[modelo], AFabricar, ADeuda);

}

printf("\*\*\*Pedido de autos\*\*\*\n");

printf("Por favor ingrese el numero del modelo de auto que desea comprar. Para salir, persione 0 \n");

scanf("%i",&modelo);

}

//Cierre del programa e impresion de segundo lote

printf("\*\*\*Cierre del programa\*\*\*\n");

printf("\*\*\*Impresion de listado 2: piezas\*\*\*\n");

for(i=0;i<cantidadPieza;i++){

//Corroboración de piezas a comprar

if(VPCP[i]>VSFP[i]){

PiezasAComprar[i]=VPCP[i]>VSFP[i];

}else{

PiezasAComprar[i]=0;

}

//Impresión del segundo listado

printf("\n De la pieza %i\n el stock inicial fue de %i \n el stock final es de %i \n y la cantidad de piezas a comprar es de %i", i+1, VSIP[i], VSFP[i], PiezasAComprar[i]);

}

}

return 0;

}