

Casos de Estudio

CE10A: Analizar el siguiente algoritmo modular en pseudocódigo que mediante un menú de opciones resuelve:

1. Cargar el vector G con F datos enteros.
2. Mostrar el vector G.
3. Agregar un dato X al vector G.
4. Modificar un elemento del vector G con un nuevo valor X en la posición P.
5. Insertar un elemento X en el vector G en la posición P.
6. Eliminar el elemento de la posición P del vector G.
7. Salir

```
procedimiento Eliminar(V,N,P)
[V: tvector pasado por referencia]
[N: entero pasado por referencia]
[P: entero pasado por valor]
variable
i: entero
inicio
Para i desde P Hasta N-1 Con Paso 1 Hacer
V[i] <- V[i+1]
FinPara
N <- N - 1
Fin //Eliminar
```

```
procedimiento Insertar(V,N,X,P)
[V: tvector pasado por referencia]
[N: entero pasado por referencia]
[X: entero pasado por valor]
[P: entero pasado por valor]
variable
i: entero
inicio
Para i desde N Hasta P Con Paso -1 Hacer
V[i+1] <- V[i]
FinPara
V[P] <- X
N <- N + 1
Fin //Insertar
```

```
procedimiento Modificar(V,N,X,P)
[V: tvector pasado por referencia]
[N: entero pasado por valor]
[X: entero pasado por valor]
[P: entero pasado por valor]
inicio
V[P] <- X
Fin //Modificar
```



```
procedimiento Agregar (V,N,X)
[V: tvector pasado por referencia]
[N: entero pasado por referencia]
[X: entero pasado por valor]
inicio
N <- N + 1
V[N] <- X
Fin //Agregar
```

```
procedimiento Mostrar(V, N)
[V: tvector pasado por valor]
[N: entero pasado por valor]
variable
i: entero
inicio
Escribir 'Mostrar los datos del vector'
Para i desde 1 Hasta N Con Paso 1 Hacer
Escribir 'V[' , i , ']=' , V[i]
FinPara
Fin //Mostrar
```

```
procedimiento Cargar(V,N)
[V: tvector pasado por referencia]
[N: entero pasado por referencia]
variable
i: entero
inicio
Escribir 'Ingresar la cantidad de datos del vector'
Leer N
Para i desde 1 Hasta N Con Paso 1 Hacer
V[i] <- Aleatorio(100,999)
FinPara
Fin //Cargar
```

```
procedimiento Menu(Op)
[Op: entero pasado por Referencia]
inicio
Escribir 'Menu de opciones'
Escribir '1. Cargar el vector'
Escribir '2. Mostrar el vector'
Escribir '3. Agregar un valor X'
Escribir '4. Modificar por X en la posición P'
Escribir '5. Insertar X en la posición P'
Escribir '6. Eliminar el elemento de la posición P'
Escribir '7. Salir'
Escribir 'Elija una Opción'
Leer Op
Fin //Menu
```

```
Algoritmo CE10A
tipo
tvector=arreglo[1..100] de entero
variable
F,X,P,Op: entero
G: tvector
inicio
Repetir
Menu(Op)
Segun Op Hacer
1: Cargar(G,F)
2: Mostrar(G,F)
3: Escribir 'Ingrese X'
Leer X
Agregar(G,F,X)
4: Escribir 'Ingrese X y P'
Leer X, P
Modificar(G,F,X,P)
5: Escribir 'Ingrese X y P'
Leer X, P
Insertar(G,F,X,P)
6: Escribir 'Ingrese P'
Leer P
Eliminar(G,F,P)
7: Escribir 'adios'
De Otro Modo
Escribir 'Error'
Fin Segun
Hasta Que Op = 7
Fin
```

		TRABAJO PRÁCTICO 10 CICLO LECTIVO 2019	MODULARIZACIÓN VECTORES
--	--	---	--

CE10B Hacer un algoritmo en pseudocódigo y/o Pseint que elimine todos los valores K de un vector V con N datos.

Ejercicios para la clase Práctica

1. Hacer un módulo denominado **ValorCentral** que tiene dos parámetros formales V y Fila. El primer parámetro V es un vector de números reales y el segundo parámetro Fila es de tipo entero y representa la longitud del vector V. El módulo debe devolver un valor real de acuerdo a la siguiente consideración: si la cantidad de elementos de V es impar devuelve el elemento central; en caso contrario, devuelve el promedio de los valores centrales.
2. Hacer un módulo denominado **CopiaVector** que tiene tres parámetros formales A, B, y N. Los parámetros A y B son vectores de números enteros y el tercer parámetro N es de tipo entero y representa la cantidad de datos que tiene el vector A. El módulo debe devolver en el parámetro B la copia de los datos del vector A.
3. Hacer un módulo denominado **BuscarPrimerImpar** que tiene dos parámetros formales V y N. El primer parámetro V es un vector de números enteros y el segundo parámetro N es la longitud del vector. El módulo debe devolver la posición del primer número impar del vector, pero si el vector no tiene ningún número impar entonces debe devolver el valor cero.
4. Una playa de estacionamiento de un supermercado solicita una aplicación que le permita gestionar el movimiento de vehículos durante una jornada. El programa debe contar con las siguientes opciones:
 1. Ingresar un vehículo solicitando su número de patente (ver consideraciones para el ingreso).
 2. Buscar un vehículo e indicar en que número de box se encuentra.
 3. Realizar la salida de un vehículo de la playa de estacionamiento. El proceso implica buscar un vehículo y dejar vacío el box donde se encontraba.
 4. El menú principal de la aplicación debe contar con dos alertas actualizadas permanentemente, que indican la disponibilidad de boxes para dos tipos de vehículos, por ejemplo:
 - Disponibilidad Autos: **4**
 - Disponibilidad Camionetas: **0**
 5. Listar los vehículos ubicados en la playa de estacionamiento discriminándolos por tipo de vehículo.

Consideraciones para el ingreso

- La capacidad máxima del estacionamiento es de 25 boxes, 15 para automóviles y 10 para camionetas
- Al inicio de la jornada todos los boxes se encuentran vacíos
- Los boxes del 1 al 10 se encuentran en planta baja y son solo para camionetas
- Los boxes del 11 al 25 se encuentran en la segunda planta y son solo para automóviles.
- Todos los vehículos se estacionan en el primer lugar que encuentren libre siempre y cuando estén en la sección que les corresponda (camionetas en planta baja y autos en segunda planta).