# UT3-TA1

## Ejercicio 1

### Consumo de variables:

→ Nombre del producto: String → 2×(N° de char) + 38 bytes
→ Código de identificación: String → 2×(N° de char)+38 bytes
→ Precio unitario: Float → 4 bytes

### Peor de los casos:

Nombre de producto: 30 char máximo → 2×30+38 = 98
Código de identificación: 20 char máximo → 2×20+38 = 78
Precio unitario: 4 bytes → 4

} 180 bytes

### Ventajas y desventajas de arrays y listas encadenadas:

#### Arrays

★ Ventajas
- Acceso rápido a los elementos por índice
- Menos sobrecarga de memoria por punteros

★ Desventajas:
- Tamaño fijo: si se llena hay que redimensionar, lo cual es costoso
- Se desperdicia memoria en caso de baja ocupación

#### Listas Encadenadas:

★ Ventajas
- Tamaño dinámico: Crece y decrece según las necesidades
- Facilita la inserción y eliminación de elementos

★ Desventajas:
- Acceso mas lento: Para buscar un elemento, hay que recorrer los nodos (con costo)
- Mayor uso de memoria debido a los punteros

Caso - Suponemos 10 productos

| Array | Lista |
| --- | --- |
| Memoria objeto producto<br>180 bytes x 10 objetos + 12 bytes del head del array<br><br>1812 bytes en total en Objetos | Memoria objeto producto<br>184 bytes x 10 objetos + 24 por lista encadenada + 16 + 220 por nodo que contiene la referencia al nodo y el objeto + 40<br><br>2140 bytes en total en objetos |

Observacion: Creando un array de tamaño justo es mas efectivo respecto a las listas. El problema es cuando se guarda memoria sin usar

Respuesta pregunta 1:

Costo memoria array: El costo es menor pero no es conveniente si no se sabe exactamente cuantos objetos se van a necesitar

Costo memoria lista encadenada: El costo es mayor pero es mas eficiente ya que no es mas facil insertar o eliminar los objetos

Respuesta pregunta 2:

El problema con el array sería la redimension del mismo que implicaría mucho tiempo en pasar los productos viejos al nuevo array mas grande.

Si se elimina un producto porque se deja de vender habría que correr todos los productos un lugar en el array

Con las listas no existe este problema ya que las listas se pueden redimensionar sin necesidad de pasar los productos a una nueva lista

Respuesta pregunta 3:

Array: es solo eficiente si se sabe exactamente cuantos productos se van a vender (y que ninguno se deje de vender)

Lista: Es menos eficiente en terminos de costo pero mas eficiente a la hora de agregar o eliminar productos (no hay que redimensionar)

# UT3-TA1

## Ejercicio 2

Precondición:

- El nodo pasado para agregar a la lista, no es null

Post condición:

- La lista tendra un nuevo nodo insertado, al inicio.

## Algoritmo en pseudocodigo:

```
agregarArticulo (articulo)
  Comienzo
      Si (lista ! null) Entonces
         articulo siguiente ← lista
         lista ← articulo
      Sino
         lista ← articulo
      FinSi
```

## Descripción en lenguaje natural

El metodo recibe el nodo con la informacion lista para insertar en la lista. La inserción es al inicio.
Si la lista es vacía la lista queda apuntando al artículo, de lo contrario el siguiente de artículo queda apuntando al que estaba anteriormente en la primera posición. Finalmente la lista queda apuntando al nuevo articulo