### Paso1: Lenguaje Natural:

Se necesita tener una lista encadenada simple que contenga los productos de una empresa, ordenados de acuerdo a su número de código, de forma ascendente (el modelo “lista encadenada simple”, un concepto conocido o manejado por todos).

Cada producto será un objeto componente de un elemento “Nodo” (concepto también conocido por todos), el cuál será ingresado a la lista.

Para hacerlo se verificará inicialmente que la lista no se encuentre vacía y de ser así, se ingresará el nuevo Nodo en el primer lugar de la misma.

De no encontrarse vacía, se verificará que el número de código del nuevo elemento sea menor al primer elemento de la lista, en cuyo caso se ingresará el nuevo Nodo en el primer lugar de la lista.

En caso contrario, se verificará que el código sea mayor al del último elemento de la lista y de ser así se ingresará en el último lugar de la lista.

Por último, de no verificarse los casos anteriores se recorrerá la lista hasta encontrar en elemento cuyo código sea menor y otro a continuación que sea mayor o igual al del nuevo nodo, ingresándose este último entre los mismos.

Cada vez que se realice un ingreso, se incrementará un contador que indicará la cantidad de elementos de la lista.

### Paso2: Seudocódigo formalizado:

Decidimos implementar la lista sobre una lista encadenada simple. Para ello se utilizarán Nodos cuya carga serán objetos de la clase Producto.

**PRECONDICIONES:**

* La lista encadenada “Lista”, pudiendo encontrarse la misma sin ningún elemento.
* Los Nodos “PrimerElemento” y “UltimoElemento” referenciando a los Nodos que se encuentran en la primera y última posición de la lista, o en caso de encontrarse vacía, con valores null.
* La variable “largoLista” indicando la cantidad de elementos que tiene la lista.

**POSTCONDICIONES:**

* La “Lista” contendrá un nuevo elemento, ordenada de forma ascendente de acuerdo a su número de código.
* La variable “largoLista” se incrementará en una unidad.

**ALGORITMO EN SEUDOCÓDIGO:**

insertarOrdenado(nuevoNodo)

Comienzo

Nodo temporal ←primerNodo

SI <primerNodo = null> ENTONCES

primerNodo ← nuevoNodo;

ultimoNodo ← nuevoNodo;

largoLista ← largoLista+1;

SINO

SI <nuevoNodo.codigo <= primerNodo.codigo>ENTONCES

<nuevoNodo.siguiente ←primerNodo

primerNodo ←nuevoNodo

largoLista ←largoLista + 1>

SINO SI <nuevoNodo.codigo >= ultimoNodo.codigo>ENTONCES

<ultimoNodo.siguiente ←nuevoNodo

ultimoNodo ←nuevoNodo

largoLista ←largoLista + 1>

SINO

< Nodo temporal1 ←primerNodo.siguiente

MIENTRAS<temporal <>null>HACER

<SI<temporal.codigo < nuevoNodo.codigo AND nuevoNodo.codigo <= temporal1.codigo> ENTONCES

temporal.siguiente←nuevoNodo

nuevoNodo.siguiente←temporal1

largoLista←largoLista + 1

SALIR>

FINSI

temporal←temporal.siguiente

temporal1←temporal1.siguiente>

FINMIENTRAS

FINSI

FINSI