PROGRAMACION II 2013

Universidad Católica del Uruguay

Listas, Pilas y Colas

Listas

- Secuencia de cero o más elementos de un tipo determinado
- Sucesión de elementos separados por comas:
  - $a_1, a_2, ..., a_n$ donde n >= 0 y a_i es del tipo de elemento.
- n = 0 significa lista vacía
- a_1 es el primer elemento y a_n es el último
- a_i está en la posición i
- a_{i-1} precede a a_i , y a_i sucede a a_{i-1}
- los elementos pueden estar ordenados en forma lineal de acuerdo a sus posiciones en la lista

Listas – operaciones básicas

- ObtenerPrimero: Tipo Elemento - devuelve el primer elemento de una lista; si la lista está vacía, devuelve NULO.
- ObtenerUltimo: Tipo Elemento - devuelve el último elemento de una lista; si la lista está vacía, devuelve NULO.
- Colocar(elementoAInsertar, elementoAnterior): Inserta en la lista el "elementoAInsertar" después del “elementoAnterior”
- InsertarAlPrincipio (unElemento). Inserta unElemento al principio de la lista
- InsertarAlFinal (unElemento). Inserta unElemento al final de la lista
- EsVacia.
- BuscarElemento (unElemento). Retorna si unElemento existe o no en la lista
- BuscarEtiqueta(unaEtiqueta) Busca el TElemento que tenga la Etiqueta y si lo encuentra, retorna el elemento; de lo contrario retorna NULO .
- Eliminar (unElemento) – elimina el elemento indicado de la lista, devolviendo TRUE si la eliminación fue exitosa o FALSE en caso contrario.
- ImprimirEtiquetas
- Siguiente(unElemento);
- Anterior(unElemento);

Listas – operaciones avanzadas

Universidad Católica del Uruguay

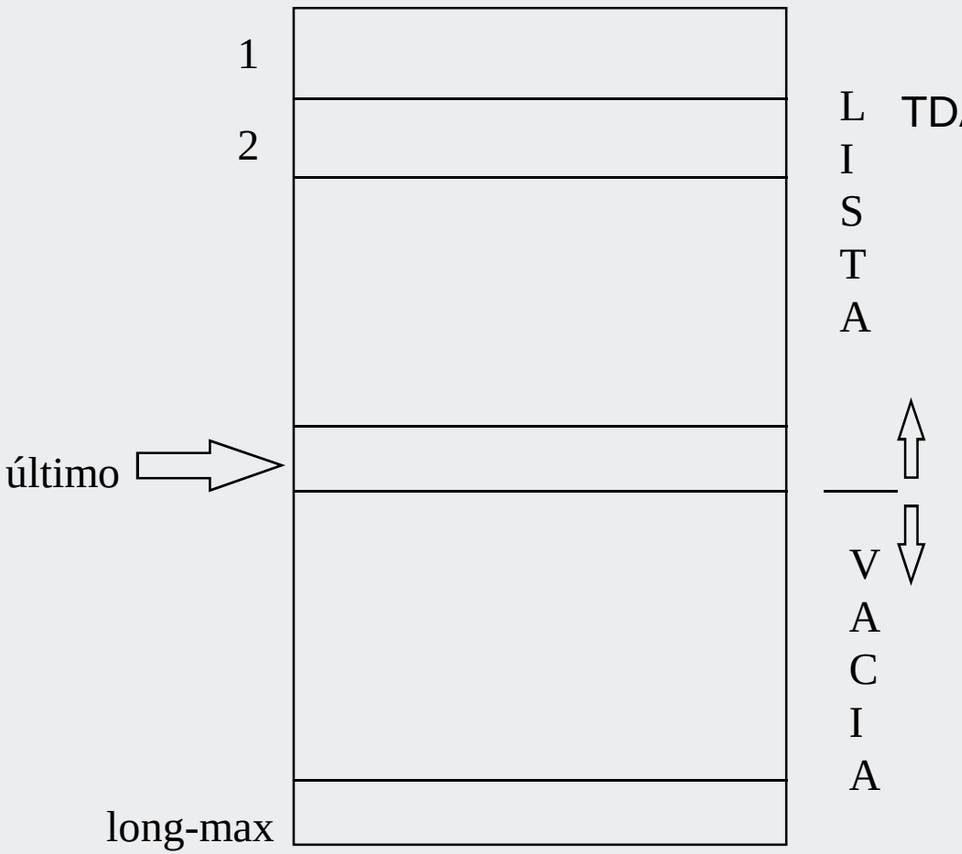
- Eliminar (unaEtiqueta). Elimina de la lista el primer TElemento cuya Etiqueta sea unaEtiqueta, y devuelve *true* si lo hizo o *false* si no había tal elemento en la lista
- ObtenerCantidadElementos;
- InsertarOrdenado (unElemento).
- Purgar (o Vaciar)
- Invertir. Devuelve una nueva Lista invertida
- Mezclar (otraLista) Devuelve una nueva Lista mezclada
- QuitarPrimero. Devuelve el primer Elemento de la lista, quitándolo de la misma

Tipo Elemento y Tipo Etiqueta

Universidad Católica del Uruguay

- Tipo Elemento
  - Atributos:
    - Siguiente : TipoElemento
    - Etiqueta: Tipo Etiqueta
  - Operaciones
    - ImprimirEtiqueta
- Tipo Etiqueta :
  - Atributos
    - Etiqueta
  - Operaciones:
    - Imprimir

Implementación con arreglo



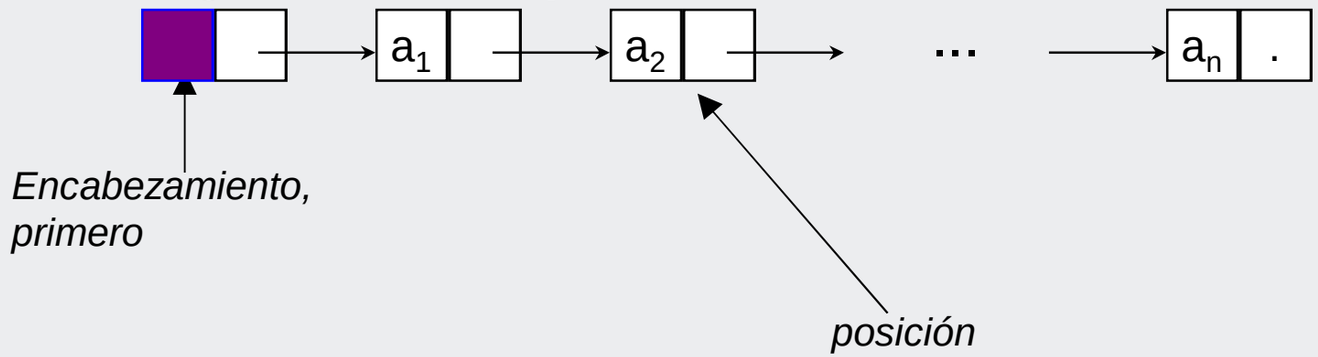

```
TDA Lista
    elementos : array [1..longMax]
                    de  tipoElemento;
    último : integer;
```

Implementación con referencias

- TDA TipoElemento

  etiqueta : TipoEtiqueta;

  siguiente : TipoElemento;



- TDA Lista

  Primero: TipoElemento;

Implementación con referencias

Universidad Católica del Uruguay

- Evita el empleo de memoria contigua.
- Evita los desplazamientos de elementos al insertar o eliminar.
- Precio adicional : espacio requerido por las referencias.
- La lista está formada por nodos o celdas; cada celda contiene un elemento de la lista y una referencia a la siguiente celda.
- La última celda contiene una referencia nula o vacía
- Existe una celda de encabezamiento que apunta a la primer celda de la lista, y no tiene ningún elemento.
- Opcionalmente podríamos tener una referencia al último nodo o celda- resulta útil

Listas: Comparación de métodos.

Universidad Católica del Uruguay

- La realización con arreglos exige especificar el tamaño máximo de la lista: se desperdicia espacio de almacenamiento.
- Ciertas realizaciones son más lentas en una representación que en otra.
- La realización con referencias utiliza para los elementos sólo el espacio real requerido por ellos, pero debe agregarse el espacio necesario para cada referencia.

Listas doblemente encadenadas

Universidad Católica del Uruguay

- En algunos casos es necesario recorrer la lista en ambos sentidos.
- Utilizamos entonces dos punteros: uno a la celda siguiente y otro a la anterior.
- Algunas operaciones son más complicadas.
- Espacio adicional para el nuevo puntero.

Operaciones en Listas, seudocódigos (1 de 2)

Universidad Católica del Uruguay

- **ObtenerPrimero**: Tipo Elemento - devuelve el primer elemento de una lista; si la lista está vacía, devuelve NULO.
- **ObtenerUltimo**: Tipo Elemento - devuelve el último elemento de una lista; si la lista está vacía, devuelve NULO.
- **Colocar (**elementoAInsertar, elementoAnterior): Inserta en la lista el "elementoAInsertar" después del “elementoAnterior”
- **InsertarAlPrincipio** (unElemento). Inserta unElemento al principio de la lista
- **InsertarAlFinal** (unElemento). Inserta unElemento al final de la lista
- **EsVacia**.
- **BuscarElemento** (unElemento). Retorna si unElemento existe o no en la lista
- **BuscarEtiqueta** (unaEtiqueta) Busca el TElemento que tenga la Etiqueta y si lo encuentra, retorna el elemento; de lo contrario retorna NULO .
- **Eliminar** (unElemento) – elimina el elemento indicado de la lista, devolviendo TRUE si la eliminación fue exitosa o FALSE en caso contrario.

Operaciones en Listas, seudocódigos (2 de 2)

Universidad Católica del Uruguay

- Eliminar (unaEtiqueta). Elimina de la lista el primer TElemento cuya Etiqueta sea unaEtiqueta, y devuelve TRUE si lo hizo o FALSE si no había tal elemento en la lista
- ObtenerCantidadElementos;
- InsertarOrdenado (unElemento).
- Purgar (o Vaciar)
- Invertir Devuelve una nueva Lista invertida
- Mezclar (otraLista) Devuelve una nueva Lista mezclada
- QuitarPrimero. Devuelve el primer Elemento de la lista, quitándolo de la misma,

Listas: Planteo de un problema 1

Universidad Católica del Uruguay

- Se necesita imprimir la etiqueta de cada elemento de una lista en el orden en el que aparecen del primero al último elemento.
- Primer paso: elaborar una solución en lenguaje natural sobre el modelo matemático lista. Escribir la solución.
- Segundo paso: refinar la solución del primer paso y elaborar un algoritmo en seudocódigo sobre el TDA LISTA.
- Tercer paso: implantar el TDA con estructuras de datos y procedimientos en un lenguaje de programación.

Resolución del Problema: PASO 1

Universidad Católica del Uruguay

- Sobre la base del modelo matemático LISTA (secuencia de elementos), una solución puede ser la siguiente:
- Voy al primer elemento, y si existe, imprimo su etiqueta.
- Avanzo hasta el segundo elemento repitiendo la operación.
- Sigo avanzando e imprimiendo hasta que llego al último elemento.

LISTAS: Algoritmos de ejemplo con seudocódigo TDA

Universidad Católica del Uruguay

- Lista.ImprimirEtiquetas

```
//imprime las etiquetas de todos los elemento, del primero al último.
//se define la operación TEtiqueta.Imprimir que imprime el contenido de una
    etiqueta.
UnElemento de Tipo Elemento

COMIENZO
UnElemento  ß  UnaLista.Primero
Mientras UnElemento <> nulo hacer
    UnElemento.Etiqueta.Imprimir
    UnElemento ß  UnElemento.Siguiente
Fin mientras
FIN
```

Seudocódigo

- PRECONDICIONES
  - **Estado** en que ***debe*** encontrarse el objeto ***antes*** de comenzar el algoritmo
  - Lenguaje natural
  - Especificación rigurosa
  - Mapeo con el código!!!
  - Ayudan a escribir los casos de prueba
- POSTCONDICIONES
  - **Estado** en que ***ha de quedar*** el objeto ***después*** de terminar el algoritmo
  - **Facilitan** la escritura de los casos de prueba.

LISTAS: Planteo de un problema 2

Universidad Católica del Uruguay

- Se necesita diseñar un algoritmo que elimine los elementos con etiqueta duplicada que puedan existir en una lista.
- Primer paso: elaborar una solución en lenguaje natural sobre el modelo matemático lista. Escribir la solución.
- Segundo paso: refinar la solución del primer paso y elaborar un algoritmo en seudocódigo sobre el TDA LISTA.
- Tercer paso: implantar el TDA con estructuras de datos y procedimientos en un lenguaje de programación.

Resolución del Problema: PASO 1

Universidad Católica del Uruguay

- Sobre la base del modelo matemático LISTA (secuencia de elementos), una solución puede ser la siguiente:
- Veo cual es el primer elemento y me voy fijando a partir del siguiente elemento y hasta el último a ver si hay alguno que tenga la misma etiqueta; en caso afirmativo elimino ese duplicado.
- Avanzo hasta el segundo elemento repitiendo la operación.
- Sigo avanzando y verificando hasta que llego al último elemento. (Comparamos cada uno con todos los demás)

LISTAS: Algoritmos de ejemplo con seudocódigo TDA

Universidad Católica del Uruguay

- UnaLista.EliminaDuplicados

```
//elimina los elementos con etiqueta duplicada de la lista
UnElemento, ElOtroElemento de Tipo Elemento

COMIENZO
UnElemento  ß  UnaLista.Primero
Mientras UnElemento <> nulo hacer
      ElOtroElemento  ß  UnElemento.Siguiente
      mientras ElOtroElemento <> nulo hacer
        si ElOtroElemento.Etiqueta = UnElemento.Etiqueta
                    entonces  UnaLista.Elimina(ElOtroElemento)
                    si no ElOtroElemento  ß  ElOtroElemento.Siguiente
        fin si
      fin mientras
      UnElemento ß  UnElemento.Siguiente
Fin mientras
FIN
```

TDA y LISTAS EN LA WEB

Universidad Católica del Uruguay

- http://en.wikipedia.org/wiki/Abstract_data_type#Defining_an_ADT
- http://www.answers.com/topic/abstract-data-type#Typical_operations
- http://es.wikipedia.org/wiki/Tipo_de_dato_abstracto

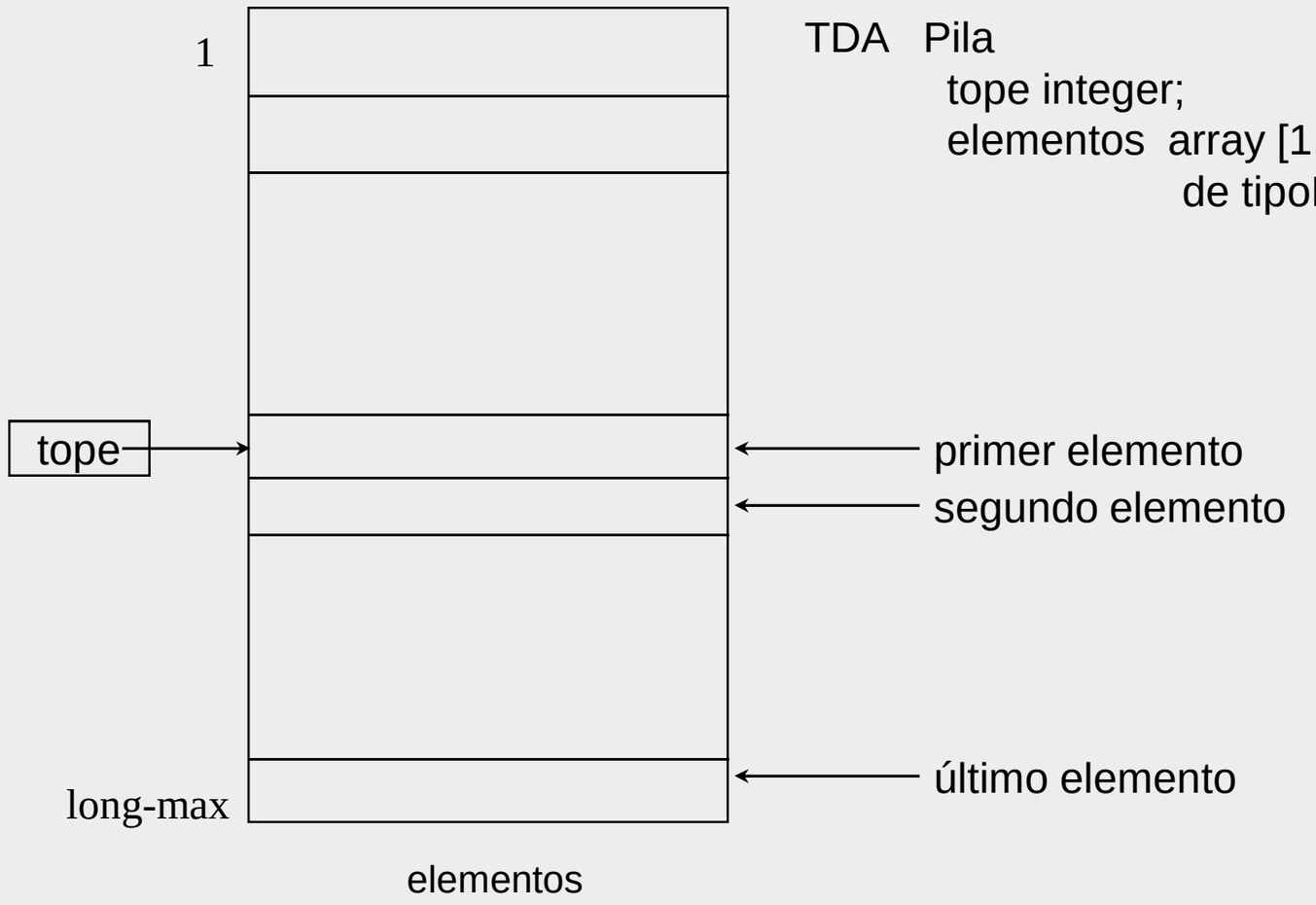
- http://en.wikipedia.org/wiki/Linked_list
- http://es.wikipedia.org/wiki/Lista_(estructura_de_datos)
- http://en.wikipedia.org/wiki/List_(computing)

Pilas

Universidad Católica del Uruguay

- Tipo especial de lista en que todas las inserciones y eliminaciones se hacen en el mismo extremo, denominado tope.
- Listas LIFO (Last In First Out).
- Operaciones normales para el TDA Pila:
  - Anula
  - Tope
  - Saca
  - Mete(unElemento)
  - Vacía

REALIZACION DE PILA CON ARREGLO




```
TDA   Pila
        tope integer;
        elementos  array [1..longMax]
                        de tipoElemento;
```

primer elemento

segundo elemento

último elemento

Operaciones en Pilas

Universidad Católica del Uruguay

- Anula: : vaciar la pila
- Tope: devuelve el elemento que está en el tope sin quitarlo
- Saca : devuelve el elemento del tope, quitándolo de la pila
- Mete(unElemento): pone un elemento en el tope de la pila, si puede
- Vacía: indica si la pila está vacía o no

Colas

Universidad Católica del Uruguay

- Tipo especial de lista en el que los elementos se insertan en un extremo (el posterior) y se extraen por el otro (el anterior o frente).
- Listas FIFO (First In First Out).
- Operaciones análogas a las de las pilas:
  - Anula
  - Frente
  - PoneEnCola(unElemento)
  - QuitaDeCola
  - Vacía

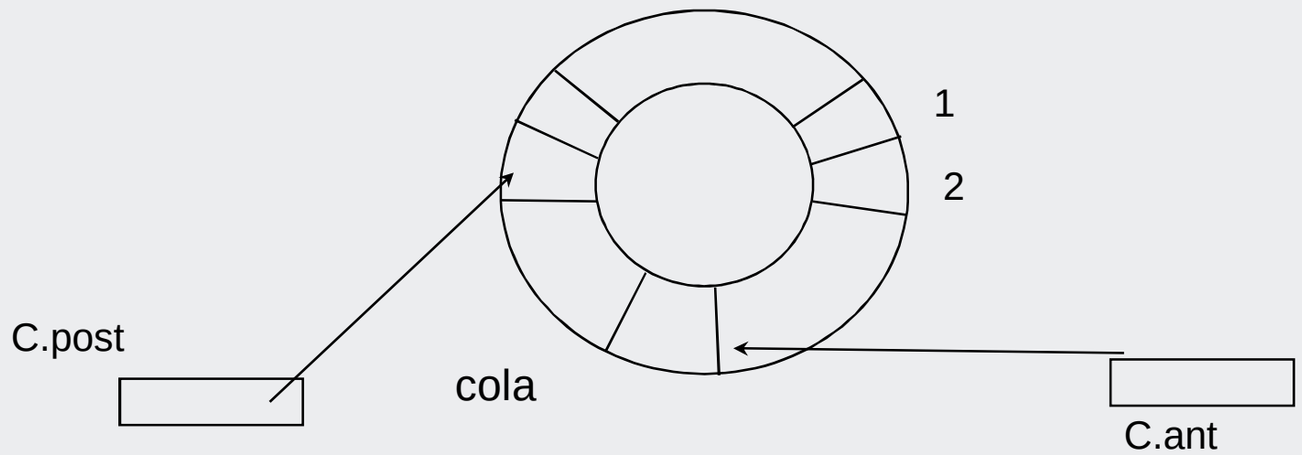
Realización de colas con referencias

Universidad Católica del Uruguay

- Igual que en las pilas, cualquier operación de listas es válida para las colas.
- Se puede mantener una referencia al principio y otra al final de la cola.
- Se puede utilizar como encabezamiento una celda adicional en la cual se ignora el campo elemento.

Implementación de colas con arreglos “circulares”

- La representación de listas por medio de arreglos puede también usarse para las colas, pero no es muy eficiente.
- Para mejorar, utilizamos un arreglo como si fuera un círculo, es decir, después del último elemento viene nuevamente el primero.
- La cola se encuentra en alguna parte del círculo, utilizando posiciones consecutivas.



Otros comportamientos interesantes de las listas

Universidad Católica del Uruguay

- Invertir
- Ordenar
- Mezclar dos listas

- Estos comportamientos deben implementarse en el código JAVA del TDA

LISTAS: Planteo de un problema 3

Universidad Católica del Uruguay

- Se necesita Ordenar una lista.
  - A partir de una lista de entrada, obtener una segunda lista en la cual los elementos de la primera estarán ordenados de menor a mayor (de acuerdo a sus claves)
- Primer paso: elaborar una solución en lenguaje natural sobre el modelo matemático lista. Escribir la solución. Indicar Precondiciones, Postcondiciones y todas las posibles situaciones internas.
- Segundo paso: refinar la solución del primer paso y elaborar un algoritmo en seudocódigo sobre el TDA LISTA.
- Tercer paso: implantar el TDA con estructuras de datos y procedimientos en un lenguaje de programación.

Resolución del Problema: PASO 1

Universidad Católica del Uruguay

- Sobre la base del modelo matemático LISTA (secuencia de elementos), una solución puede ser la siguiente – en lenguaje natural - :
- Si la lista origen no está vacía
  - Para cada elemento de la lista de origen
    - Extraerlo de la lista origen
    - Insertarlo en la lista de salida, en forma ordenada
  - Devolver la lista de salida
- Precondiciones?
- Postcondiciones?
- Situaciones posibles en el algoritmo?

Resolución del Problema: PASO 2

Universidad Católica del Uruguay

- Algoritmo

Ejercicios, seudocódigo y práctico en papel

Universidad Católica del Uruguay

- Escriba un algoritmo para intercalar dos listas clasificadas.
- Se define el palíndromo de una palabra a la palabra obtenida invirtiendo todas las letras; por ejemplo el palíndromo de “arroz” es “zorra”. Escriba un algoritmo no recursivo que dada una palabra, genere su palíndromo. El algoritmo debe ser escrito sobre la base de un TDA PALABRA, con sus primitivas correspondientes.
- Escriba un procedimiento para intercambiar dos elementos consecutivos cualesquiera de una lista, en la cual denotaremos como elemento al primero de ellos.
- Escriba un algoritmo para invertir una lista simple.
- Desarrolle los algoritmos para implementar las operaciones de Union e Intersección sobre el TDA LISTA, utilizado para representar un CONJUNTO. Las listas de entrada se encuentran ordenadas