

INVERTIR UN ARRAY

Algoritmo InvertirArray(A, i, j)

Entrada: Un array A de enteros e indices enteros no negativos i y j

Salida: Los elementos de A entre los indices i y j, invertidos

COM

Si $i < j$ then

Intercambiar $A[i]$ y $A[j]$

InvertirArray(A, $i+1$, $j-1$)

FINSI

FIN

InvertirArrayIterativo(A, i, j)

COM

Mientras $i < j$ do

INTERCAMBIA $A[i]$ con $A[j]$

$i = i + 1$

$j = j - 1$

FIN MIENTRAS

FIN

RECURSION BINARIA

- $F_0 = 0$
- $F_1 = 1$
- $F_i = F_{i-1} + F_{i-2} \quad \forall i > 1$

Algoritmo FibonacciBinario (k)

Entrada: entero k no negativo

Salida: el k -esimo numero de Fibonacci

COM

Si $k=0$ o $k=1$

devolver k

SINO

devolver $\text{FibonacciBinario}(k-1) +$
 $\text{FibonacciBinario}(k-2)$

FINSI

FIN

USANDO RECURSION LINEAL

Algoritmo FibonacciLineal(k)

Entrada: Un entero no negativo k

Salida: Par de números de Fibonacci(F_k, F_{k-1})

COM

Si $k = 1$

devolver ($k, 0$)

SINO

$(i, j) = \text{FibonacciLineal}(k - 1)$

devolver ($i + j, i$)

FIN

- Ejecuta en tiempo $O(k)$

RECURSIÓN

TDALista. Invertir

COM

Si (no vacía) entonces

$x \leftarrow \text{Primer}o$

Eliminar (Primer)

Invertir

Inserir Ultimo (x)

FINSI

FIN

ARRAYS

El siguiente algoritmo encuentra el lugar LUG y el valor MAX del elemento mayor del array DATOS.

1. Inicializar $k := 1$, $LUG := 1$ y $MAX := DATOS[1]$
2. Repetir los pasos 3 y 4 mientras $k \leq N$:
3. Si $MAX < DATOS[k]$ entonces
 $LUG := k$ y $MAX := DATOS[k]$
- FIN SI
4. $k := k + 1$
- FIN CICLO PASO 2
5. Escribir LUG , MAX
6. Salir

IMPLEMENTACION del TDA LISTA con referencias

• TDA Lista

- Atributos:

- Primero: Tipo NodoLista;

- Operaciones:

- Insertar (E de TipoElemento, pos de TipoNodoLista): TipoElemento

- Eliminar (pos de TipoNodoLista): TipoElemento

- Buscar (x de TipoEtiqueta): TipoNodoLista

• TDA NodoLista

- Atributos:

- siguiente: TipoNodoLista

- elementos: TipoElemento

- etiqueta: TipoEtiqueta

- Operaciones:

- ImprimirEtiqueta

- TDA Etiqueta

- Atributos:

- Etiqueta

- Operaciones:

- Imprimir

- TDA Elemento

- Atributos:

- Etiqueta: TipoEtiqueta

- Datos: TipoDatos

IMPRIMIR ETIQUETAS DE LISTA

Lista. Imprimir Etiquetas

nodoActual de TipoNodo Lista
COMIENZO

nodo Actual ← UnaLista. Primero
mientras nodo Actual <> null hacer
nodoActual. Etiqueta. Imprimir
nodoActual ← nodoActual. siguiente

FIN MIENTRAS

FIN

BORRAR DUPLICADO EN LISTA

COMIENZO

nodoActual ← UnaLista. Primero

Mientras nodoActual <> null hacer

otroNodo ← nodoActual . siguiente

Mientras otroNodo > null hacer

Si otroNodo. Etiqueta = nodoActual. Etiqueta
entonces UnaLista. Elimina(otroNodo)

SINO

otroNodo ← otroNodo . siguiente

FINSI

FIN MIENTRAS

nodoActual ← NodoActual . siguiente

FIN MIENTRAS

FIN

ORDENAR LISTA

COM

Lista2 \leftarrow nuevaLista

Mientras no (vacia()) hacer

elementosActual \leftarrow Eliminar(primer)

Lista2. InsertarOrdenado(elementosActual)

FINMIENTRAS

devolver lista2

FIN

InsertarOrdenado(de tipoElemento el Elemento)

COM

v1 Nodo \leftarrow nuevoNodo(el Elemento)

Si vacia() entonces

primeros \leftarrow v1Nodo

salir

FINSI

Si $v_nodo.clave < primero.clave$
 $v_nodo \leftarrow v_nodo.sigiente$. $siguiente \leftarrow primero$
 $primero \leftarrow v_nodo$
salir

FINSI

$tempNodo \leftarrow primero$

Mientras ($tempNodo.sigiente \neq null$ y
 $tempNodo.sigiente.clave >$
 $v_nodo.clave$) hacer

$tempNodo \leftarrow tempNodo.sigiente$

FINMIENTRAS

$v_nodo \leftarrow v_nodo.sigiente$

$tempNodo.sigiente \leftarrow v_nodo$

FIN

