Informe

Análisis y diseño de Algoritmos

1er entrega

Alumno: Attilio, Nicolas

Comision : 1

Ayudante:

En este informe, se detalla la especificación algebraica con signaturas y axiomas e implementación de los tipos LISTA y ARREGLO. Para esto se diseñó contenedores genéricos, pero en el trabajo solamente se muestra con números enteros, como modo de ejemplificación.

Especificación:

TDA LISTA

CLASS Lista[Elemento]

IMPORTS

BASIC CONSTRUCTORS inicLista,agregarLista;

EFFECTIVE

TYPE Lista;

OPERATIONS

inicLista--->Lista;

agregarLista: Lista(l) x Elemento x Nat(i):--->Lista

Pre:((i>=1) and (i<=cantElemento(l)+1));

agregarPrincipio: Lista x Elemento:--->Lista;

agregarFinal:Lista x Elemento:--->Lista;

cantElemento: Lista:--->Nat;

existeElemento: Lista x Elemento:--->Boolean;

esVacia: Lista:--->Boolean;

eliminarElemento: Lista(l) x Elemento x Nat(i):--->Lista

Pre:((i>=1) and (i<=cantElemento(l)+1));

AXIOMS l:Lista; e1,e:Elemento; j,i:Nat;

agregarPrincipio(inicLista(),e)=agregarLista(inicLista(),e,1);

agregarPrincipio(agregarLista(l,e,i),e1)=agregarLista(agregarLista(l,e,i),e1,1);

agregarFinal(inicLista(),e)=agregarLista(inicLista(),e,cantElemento+1;

agregarFinal(agregarLista(l,e,i),e1)=agregarLista(agregarLista(l,e,i),e1,cantElemento+1;

cantElemento(inicLista())=0;

cantElemento(agregarLista(l,e,i))=1+cantElemento(l);

existeElemento(inicLista())=False;

existeElemento(agregarLista(l,e,i))=True;

esVacia(inicLista())=False;

esVacia(agregarLista(l,e,i))=True;

(i==j)=>(eliminarElemento(agregarLista(l,e,i),j)=l;

(i>j)=>(eliminarElemento(agregarLista(l,e,i),j)=agregarLista(eliminarLista(l,j)i-1,e));

(i<j)=>(eliminarElemento(agregarLista(l,e,i),j)=agregarLista(eliminarLista(l,j-1),i,e));

TDA Arreglo

CLASS Arreglo[Elemento]

IMPORTS

BASIC CONSTRUCTORS inicArreglo,agregarArreglo;

EFFECTIVE

TYPE Arreglo;

OPERATIONS

inicArreglo:--->Arreglo;

agregarArreglo: Arreglo(a) x Elemento x Nat(i):--->Arreglo;

Pre:((i>=1) and (i<=cantElemento(a)+1));

longArreglo: Arreglo:--->Nat;

darElemento: Arreglo(a) x Nat(i):--->Elemento;

Pre:((i>=1) and (i<=cantElemento(l)+1));

AXIOMS a:Arreglo; j,i;Nat; e:Elemento;

longArreglo(inicArreglo())=0;

longArreglo(agregarArreglo(a,i,e))=1+longArreglo(a);

(i==j)=>(darElemento(agregarArreglo(a,i,e),j))=e;

(i>j)=>(darElemento(agregarArreglo(a,i,e),j)=darElemento(a,j));

(i<j)=>(darElemento(agregarArreglo(a,i,e),j)=darElemento(a,j-1));

Resumen

En la primera parte, en la implementación de la clase Lista, trabajo con estructura de un nodo que contiene un espacio para guardar el elemento genérico y otro para desplazarse al siguiente.

En la la segunda parte de esta entrega utilizo conceptos de la clase Lista, añadiendo un anterior para moverse al elemento que le antecedente.

El corrimiento de los nodos, los trabaje de modo iterativo para evitar que la complejidad temporal no se vea tan afectada por el tamaño de la entrada. El análisis de éstos se encuentran detallados en los .cpp de cada clase.

La implementación de estos TDA se encuentran al final de este informe, en formato .zip