class ArrayList : TDAList{

private:

object array

int n //size

int capacity

public:

bool inserta(item, posicion){ --O(n)

//1. Validar rango

//1.1 determinar si hay que hacer corrimientos

//hacer corrimientos

//insertar elemento

//n++

return true

else

return false

}

object primero(){--O(1)

si n>=1 //validar rango

return array[0]

else

return null

}

object recupera(posicion){ --O(1)

//validar el rango

Si posicion >= 1 y <=n

return array[posicion-1]

else

return null

}

bool suprime(posición){

//validar rango

Si posición >= 1 & <= n

//eliminar elemento

//determinar si hay que hacer corrimientos

//hacer corrimientos

//n—

return true

else

return false

}

posición localiza(value){

for ( pos = 1 , pos <= n, pos ++) {

si value == array[pos -1]

return pos

}

return -1

}

Siguiente/anterior

Object siguiente/anterior(posición){

//No se valida el rango porque se valida en recupera

Return recupera (posición +/- 1) 🡨 depende de si queremos siguiente o anterior

}

void imprime\_lista(){

for (pos = 0 ; pos < size; pos ++)  
 imprimir(array[pos]

}

Entero anular(){

             //Valida si esta vacia

Si n>0{

for(int i =0;i<n;i++){

//Elimina el elemento en la casilla i

Suprimir (i);//Llama el metodo suprimir

                                            }

}

Retornar 1;

}

}