TAD

|  |
| --- |
| TAD: Type |
| |  | | --- | | <T>data  Type <T>next | |
| Inv: Type >= next |
| Operaciones primitivas:  Type: <T> x Type -> Type  addNext: <T> Type x Type -> Type  counter: Type<T>-> int  sort -> Type |

|  |  |
| --- | --- |
| Type(<T> Data)  “Constructor de la clase Type, añade un dato de tipo T a el atributo data de la clase”  {pre: Type==null}  {post: Type:   |  | | --- | | <data>  “” |   } |

|  |  |
| --- | --- |
| addNext(<T> Type)  “Se añade una nueva Type al next, esta tiene que ser del mismo tipo <T> que la lista a la que se le está añadiendo. Si el next es diferente de null, se llama al siguiente elemento en la lista. Cuando se añade se usa el método orderl”  {pre: Type!=null && El generic de ambas sea igual}  {post: Type:   |  | | --- | | <data>  <next> |   } |

|  |  |
| --- | --- |
| counter (<T> Data)  “Cuenta cuantos valores hay en la lista enlazada Type”  {pre: Type !=null}  {post: Type:   |  | | --- | | <data>  <next> |   Retorna un int  } |

|  |  |
| --- | --- |
| sort ()  “Organiza la lista enlazada”  {pre: Type !=null}  {post: Type:   |  | | --- | | <data>  <next> |   } |