**LISTA LINEAR SIMPLESMENTE ENCADEADA (LLSE)**

|  |
| --- |
| ***ALGORITMO .::. inicializarLLSE*** |
| **]**  *Entrada*: LE (Tipo Lista Encadeada Estática)  IA, FA (inteiros)  *Saída*: LE (Tipo Lista Encadeada Estática)  IA, FA, IL, FL, LE.Disp (inteiros) |
| **início**  LE.Disp←1;  **início**  **Para** Ind de IA incr 1 **até** FA **faça**  LE[ind].prox←Ind+1  **Fim**  LE[IA].prox← -1  LE[FA].prox← 0  **Fim** |

|  |
| --- |
| ***ALGORITMO .::. alocarNoLLSE*** |
| *Entrada*: LE (Tipo Lista Encadeada Estática)  IA, FA, IL, FL, LE.Disp (inteiro)  *Saída*: LE (Tipo Lista Encadeada Estática)  IA, FA, IL, FL, LE.Disp (inteiro)  sucesso (lógico) |
| **início**  Ind;  **Se** (LE.Disp = -1) **então**  sucesso←falso  **Senão**  **inicio**  ind←LE.Disp  LE.disp←LE[LE.disp].prox  **Retorna**(Ind)  **fim**  **fim** |

|  |
| --- |
| ***ALGORITMO .::. inserirDepoisLLSE*** |
| *Entrada*: LE (Tipo Lista Encadeada Estática)  IA, FA, IL, FL,LE.Disp, Ipo (inteiro)  infNo (Tipo nó)  *Saída*: LE (Tipo Lista Encadeada Estática)  IA, FA, IL, FL, LE.Disp (inteiro) |
| **Início**  Ind←AlocarNoLEArr  LE[Ind].info←infNo  LE[Ind].prox←LE[Ipo].prox  LE[Ipo].prox←Ind  **Fim** |

|  |
| --- |
| ***ALGORITMO .::. liberarNoLLSE*** |
| *Entrada*: LE (Tipo Lista Encadeada Estática)  IA, FA, IL, FL,  Ipo (inteiro)  *Saída*: LE (Tipo Lista Encadeada Estática)  IL, FL (inteiro) |
| **Início**  LE[Ipo].prox = LE.Disp  LE.Disp←Ipo  **Fim** |

|  |
| --- |
| ***ALGORITMO .::.liberarNoDepoisLLSE*** |
| *Entrada*: LE (Tipo Lista Encadeada Estática)  IL, FL, IA, FA (inteiro)  Ipo (inteiro)  *Saída*: LE (Tipo Lista Encadeada Estática)  IL, FL (inteiro)  sucesso (lógico) |
| **início**  ind(inteiro)  **se** LE[Ipo].prox = 0 **então**  sucesso←falso  **senão**  ind←LE[p].prox  LE[p].prox←LE[p].prox  liberarNoLEArr(LE,ind)  **fim** |