**LISTA LINEAR COM VETOR**

|  |
| --- |
| **ALGORITMO .::. inicializarLLArr** |
| *Entrada*: IA (inteiro)  *Saída*: IL, FL (inteiro) |
| **início**  IL←FL←IA-1; **fim** |

|  |
| --- |
| **ALGORITMO .::. inserirInicioLLArr** |
| *Entrada*: LL (tipoLista); IA, FA, IL, FL (inteiro); infNo (tipoNó)  *Saída*: LL (tipoLista); IL, FL (inteiro); sucesso (lógico) |
| **início**  **Se** (IA=IL) E (FA=FL) **então**  sucesso←falso  **Senão**  **inicio**  **Se** (IL=0) **então**  IL←FL←IA  **Senão**  **Se** (IL>IA) **então**  IL←IL-1  **Senão**  **início**  **Para** Ind **de** FL **incr** -1 **até** IL **faça**  LL[Ind+1]←LL[Ind]  FL←FL+1  **fim**  LL[IL]←InfNo;  Sucesso = verdadeiro  **fim**  **fim** |

|  |
| --- |
| **ALGORITMO .::. inserirFimLLArr** |
| *Entrada*: LL (tipoLista); IA, FA, IL, FL (inteiro); infNo (Tipo nó)  *Saída*: LL (tipo lista); IL, FL (inteiro); sucesso (lógico) |
| **início**  **Se** (IA=IL) E (FA=FL) **então**  sucesso←falso  **Senão**  **inicio**  **Se** (IL=0) **então {**LISTA VAZIA**}**  IL←FL←IA  **Senão**  **Se** (FL<FA) **então {**TEM ESPAÇO ATRÁS**}**  FL←FL+1  **Senão {**DESLOCAR PARA BAIXO**}**  **início**  **para** Ind **de** IL **incr** 1 **até** FL **faça**  LL[Ind-1]←LL[Ind]  IL←IL-1  **fim**  LL[FL]←InfNo;  Sucesso = verdadeiro  **fim**  **fim** |

|  |
| --- |
| **ALGORITMO .::. inserirPosKLLArr** |
| *Entrada*: LL (tipoLista) ; IA, FA, IL, FL (inteiro); K (inteiro); infNo (tipoNó)  *Saída*: LL (tipoLista); IL, FL (inteiro); sucesso (lógico) |
| **Início**  **Se** (IA=IL E FA=FL)OU(K>FL-IL+2)OU(K≤0)OU(IL=0 E K≠1) **então**  sucesso←falso  **Senão**  **inicio**  **Se** (IL=0) **então {**LISTA VAZIA**}**  IL←FL←IA  **Senão**  **Se** (FL≠FA) **então {**DESLOCAR PARA CIMA**}**  **inicio**  **Para** Ind **de** FL **incr** -1 **até** IL+K-1 **faça**  LL[Ind+1]←LL[Ind]  FL←FL+1  **fim**  **Senão {**DESLOCAR PARA BAIXO**}**  **início**  **Para** Ind **de** IL **incr** 1 **até** IL+K-2 **faça**  LL[Ind-1]←LL[Ind]  IL←IL-1  **fim**  LL[IL+K-1]←InfNo;  Sucesso = verdadeiro  **fim**  **fim** |

|  |
| --- |
| **ALGORITMO .::. removerKLLArr** |
| *Entrada*: LL (tipoLista); IL, FL (inteiro); K (inteiro)  *Saída*: LL (tipoLista); IL, FL (inteiro); sucesso (lógico) |
| **início**  **Se** (K≤0)OU(K>FL-IL+1) **então**  sucesso←falso  **Senão**  **inicio**  **para** Ind **de** IL+K-1 **até** FL-1 **faça**  LL[Ind]←LL[Ind+1]  FL←FL-1  **Se** (FL=IL-1) **então**  IL←FL←0  Sucesso = verdadeiro  **fim**  **fim** |

|  |
| --- |
| **ALGORITMO .::. acessarKLLArr** |
| *Entrada*: LL (tipoLista); IL, FL (inteiro); K (inteiro)  *Saída*: infNo (tipoNo); sucesso (lógico) |
| **início**  **Se** (K≤0)OU(K>FL-IL+1)OU(IL=0) **então**  sucesso←falso  **Senão**  **inicio**  infNo←LL[IL+K-1]  Sucesso = verdadeiro  **fim**  **fim** |

|  |
| --- |
| **ALGORITMO .::. posValLLArr** |
| *Entrada*: LL (tipoLista); IL, FL (inteiro); valBusca (tipoValor)  *Saída*: posição (inteiro) |
| **Início**  achou←falso  posição←0  I←IL  **enquanto** (I≤FL)E(não achou) **faça**  **se** LL[I].valor = valBusca **então**  **inicio**  posição←I  achou←verdadeiro  **fim**  **senão**  I←I+1  **fim** |