**Criar Lista-**

Assertivas de entrada:

* Recebe uma identidade de lista que é um string de ate 4 caracteres válidos

Assertivas de saída:

* Se pAnt ≠ NULL, então pCorr->pAnt->pProx = pCorr ; Se pProx ≠ NULL, então pCorr->pProx->pAnt->pCorr (essas são as assertivas estruturais da lista duplamente encadeada com cabeça)
* Lista foi criada
* Os ponteiros de inicio, fim e pElem corrente apontam para o mesmo local(origem da lista)

**Obter Id Lista-**

Assertivas de entrada:

\*Valem assertivas estruturais da lista duplamente encadeadas

\*Recebe uma lista já criada (e não nula)

Assertivas de saída:

* Valem assertivas estruturais da lista duplamente encadeadas
* Ponteiro foi preenchido com identidade da lista enviada

**Inserir nó-**

Assertivas de entrada:

* Valem assertivas estruturais da lista duplamente encadeadas
* Lista existe (já foi criada, mas pode estar vazia)
* Nó corrente aponta para o último elemento da lista
* Valor a ser inserido é valido (nesse caso, é um char)
* Assertivas de saída:
* Valem assertivas estruturais da lista duplamente encadeadas
* Lista tem um novo nó, com a informação recebida, e o elemento corrente aponta para ele, bem como o ponteiro de final de lista (se a lista estava vazia esse será o primeiro nó)

**Obter Nó-**

Assertivas de entrada:

* Recebe um ponteiro para preencher qual informação do nó que o corrente aponta
* Lista já foi criada
* Valem assertivas estruturais da lista duplamente encadeadas

Assertivas de saída:

* Valem assertivas estruturais da lista duplamente encadeadas
* Ponteiro está preenchido com o valor que o nó corrente aponta

**Excluir nó corrente-**

Assertivas de entrada:

Lista existe com pelo menos três nós

Corrente aponta para o nó intermediário que precisa ser excluído

Valem assertivas estruturais da lista duplamente encadeada com cabeça

Assertivas de saída:

Valem as assertivas estruturais da lista duplamente encadeada com cabeça

Nó corrente foi excluído

Corrente aponta para o próximo

**Ir prox-**

Assertivas de entrada:

Recebe uma lista já criada

Valem as assertivas estruturais da lista duplamente encadeada com cabeça

Assertivas de saída:

Valem as assertivas estruturais da lista duplamente encadeada com cabeça

O ponteiro corrente aponta para o próximo, no caso que era o último, ele continua apontando para o mesmo elemento

**Ir ant-**

Assertivas de entrada:

* Valem as assertivas estruturais da lista duplamente encadeada com cabeça
* Recebe uma lista já criada

Assertivas de saída:

* Valem as assertivas estruturais da lista duplamente encadeada com cabeça
* O ponteiro corrente aponta para o anterior, e caso o enviado fosse o primeiro, continua apontando para o mesmo elemento

**Alterar nó corrente-**

Assertivas de entrada:

Valem as assertivas estruturais da lista duplamente encadeada com cabeça

Nó corrente aponta para um índice valido

Assertivas de saída:

Valem as assertivas estruturais da lista duplamente encadeada com cabeça

Nó corrente agora aponta para o mesmo elemento, mas ao invés de conter um índice, contém um char

**Destrói lista-**

Assertivas de entrada:

* Valem as assertivas estruturais da lista duplamente encadeada com cabeça
* Recebe uma lista criada

Assertivas de saída:

* Lista foi destruída e o espaço alocado por ela foi liberado

**Criar tabuleiro-**

Assertivas de entrada:

* Recebe o endereço de uma cabeça de matriz a ser preenchida

Assertivas de saída:

* Valem todas as assertivas de estruturais de lista
* Ponteiro agora foi preenchido para a estrutura criada, que é uma matriz composta por listas ( com 8 linhas e 8 colunas )

**Inserir peca-**

Assertivas de entrada:

* Recebe número da linha e da coluna válidos (menores ou iguais a 8)
* A posição da casa no encontro da linha com a coluna está vazia
* Tabuleiro já foi criado
* Recebe cor valida (‘P’ ou ‘B’) e identidade (Sting de 4 caracteres)

Assertivas de saída:

* Peça foi criada e inserida na posição desejada do tabuleiro

**Mover peca-**

Assertivas de entrada:

Peça já foi criada, com suas informações preenchidas ( e não é vazia )

Posição para qual ela vai se mover está vazia, além de ser uma posição valida (contida dentro do tabuleiro)

Movimento é permitido para aquela peça

Tabuleiro já foi criado

Assertivas de saída:

Paca agora está na posição desejada

Sua posição antiga agora está vazia (‘V’)

**Retirar peca-**

Assertivas de entrada:

* Existe uma peça na coordenada enviada
* Tabuleiro existe

Assertivas de saída:

* Peça foi retirada
* Posição (casa) que ela estava agora está vazia

**Obter peca-**

Assertivas de entrada:

* Tabuleiro está criado
* Posição enviada é válida
* Ponteiro a ser preenchido é do mesmo tipo estrutural que peça

Assertivas de saída:

* Ponteiro agora contem informação sobre a peça que estava na posição inserida (pode ser uma peça ou ‘V’-vazio)

**Obter lista ameaçantes-**

Assertivas de entrada:

* Recebe índice de linha e coluna válidos (menores ou igual a 8)
* Tabuleiro tem que estar criado e preenchido (com vazios ou com peças)

Assertivas de saída:

* Valem assertivas estruturais de lista
* Foi obtido a lista ameaçantes que estava na posição dos índices recebidos

**Obter lista ameaçados-**

Assertivas de entrada:

Recebe índice de linha e coluna válidos (menores ou igual a 8)

Tabuleiro tem que estar criado e preenchido (com vazios ou com peças)

Assertivas de saída:

Valem assertivas estruturais de lista

Foi obtido a lista ameaçados que estava na posição dos índices recebidos

**Destrói tabuleiro-**

Assertivas de entrada:

Recebe o endereço da cabeça do tabuleiro (já criado)

Assertivas de saída:

Tabuleiro foi destruído, e todos seus componentes também. Além disso, toda a memória alocada antes por ele e seus componentes foram liberadas

**Insere nome peça-**

Assertivas de entrada:

* Existe uma peça criada
* O nome a ser inserido é um string de até quatro caracteres

Assertivas de saída:

* Peça está preenchida (no campo do nome) com nome enviado

**Insere cor peça-**

Assertivas de entrada:

* Existe uma peça criada
* A cor é um char contendo ‘P’ ou ‘B’

Assertivas de saída:

* A peça está preenchida com a cor enviada (no campo cor)

**Obtém nome da peça-**

Assertivas de entrada:

* Recebe ponteiro para um string
* Tabuleiro e peça já foram criados

Assertivas de saída:

* Ponteiro está preenchido com o nome da peça enviada

**Obtém cor da peça-**

Assertivas de entrada:

* O ponteiro recebido é do tipo char
* Tabuleiro e peça já foram criados

Assertivas de saída:

* Ponteiro está preenchido com a cor da peça enviada

**Cria Peça-**

Assertivas de entrada:

* Recebe ponteiro para esse tipo de estrutura

Assertivas de saída:

* Foi criado a estrutura com seus respectivos campos (nome, cor e movimento- todos preenchidos com nome ‘V’, cor: NULL e movimento: NULL)