

Assertivas estruturais

Grafo:

Antecessor aponta de volta para o vértice como sucessor

Para todo vértice pVerAnt em pVer->pAntecessores existirá uma aresta pAreSuc em pVerAnt->pSucessores tal que pAreSuc->pVertice = pVer

Sucessor aponta de volta para o vértice como antecessor

Para toda aresta pAreSuc em pVer->pSucessores existirá um vértice pVerAnt em pAreSuc->pVertice->pAntecessores tal que pVerAnt = pVer

Se está na origem tem que estar na lista de vértices

Para todo vértice pVerOri em pGrafo->pOrigens existirá um vértice pVer em pGrafo->pVertices tal que pVerOri = pVer

Se tem corrente ele tem q estar na lista de vértices

Se pGrafo->pCorrente != NULL => Existirá um vértice pVer em pGrafo->pVertices tal que pVer = pGrafo->pCorrente

Todos os antecessores dos vértices de um grafo têm q estar na lista de vértices desse grafo

Para todo vértice pVer e pVerAnt tal que pVer pertence à pGrafo->pVertices e pVerAnt pertence à pVer->Antecessores existirá um pVer2 pertencente à pGrafo->pVertices tal que pVerAnt = pVer2

Todas as arestas dos vértices de um grafo G têm que apontar para um vértice que está na lista desse grafo

Para todo vértice pVer e aresta pAre tal que pVer pertence à pGrafo->pVertices e pAre pertence à pVer->pSucessores, existirá um pVer2 pertencente à pGrafo->pVertices tal que pVer2 = pAre->pVertice.

O número de elementos na lista de vértices deve ser o exato numero do campo qtdVertices contido no cabeça do grafo.

A soma de todos os dados alocados para o armazenamento do grafo deve ser igual ao valor contido no campo TotalEspaçoAlocado presente no cabeça do grafo

Lista:

pCabeca->pOrigemLista != NULL => pCabeca->pOrigemLista->pAnt = NULL

pCabeca->pFimLista != NULL => pCabeca->pFimLista->pProx = NULL

A soma de todos os dados alocados para o armazenamento da lista deve ser igual ao valor contido no campo TotalEspaçoAlocado presente no cabeça da lista

