**Funções do grafo**

* **criagrafo (Grafo \*\* grafo )**
  + cria dinamicamente uma estrutura grafo
    - cria uma lista origens vazia
    - cria uma lista vértice vazia
    - aloca uma estrutura grafo
    - insere listas na estrutura
* **criavertice ( int id , LIS\_tppLista \* Grafo );**
  + Grafo será global ou o módulo deverá permitir operações com gráficos simultâneos?
  + cria o vértice e inicializa uma lista de arestas vazia
    - checa se vértice já existe
    - cria uma lista de arestas vazia
    - cria uma estrutura vértice
    - cria uma lista vazia e insere o vértice
    - cria um novo elemento na lista vértices (grafo) e insere lista do vértice criado
    - **DÚVIDA: Criar sempre uma origem pro vértice novo?**
* **criaaresta ( int id\_vertice\_A , int id\_vertice\_B , Grafo \*\* grafo )**
  + cria uma aresta de A para B **E** de B para A
    - verificar se existem vértices A e B (encontrá-los) e conseguir seus ponteiros
      * ultilizará uma função estática buscar vértice encapsulada no módulo GRAFO
      * irá em cada elemento da lista a partir do início obtendo valor do elemento corrente (lista vértice) e obterá o valor do elemento corrente da lista vértice (estrutura vértice) e verificará de o valor da estrutura vértice == id\_vertice\_X
      * retornará o ponteiro para a cabeça da estrutura vértice
    - obtem ponteiro para elemento corrente das duas cabeças das estruturas vértices
    - confere se existe aresta de A para B
    - cria nó nas listas de arestas dos vértices obtidos contendo ponteiro para a outra cabeça
    - **DÚVIDA: Deverá existir uma função remover aresta/vértice?**
    - **DÚVIDA: Como operações como criar aresta e remover aresta afetam a lista de origens?**
* **removeraresta ( int id\_vertice\_A , int id\_vertice\_B , Grafo \*\* grafo )**
  + verifica se existe vertices A e B
  + verificar se existe aresta A->B
  + remove aresta da lista de arestas de ambos vértices
  + **DÚVIDA: (Lembrete) Como isso afeta a lista de origens?**
* **destroivertice ( int id\_vertice , Grafo \*\* grafo)**
  + verifica se vertice existe
  + percorre lista de arestas desse vertice removendo arestas para si na lista de arestas dos vértices
  + destroi vertice
* **Possíveis funções?**
  + É alcançável
  + <https://www.cs.bu.edu/teaching/c/graph/linked/>
  + Menor caminho até saída ou algum caminho até a saída?