```
void joinwt (listaDeNodos node, int p, int q, int wt) {
  int r, r2;
  /* pesquisa na lista de arcos emanando de node[p] um arco para
  node[q] */
  r2 = -1;
  r = node[p].point;
  while (r >= 0 && node[r].point != q) {
    r2 = r;
    r = node[r].next;
  }
  if (r >= 0) { /* node[r] representa um arco de node[p] para node[q] */
    node[r].info = wt;
    return;
  }
  /* um arco de node[p] para node[q] não existe. Esse arco deve ser
  criado. */
  r = getnode();
  node[r].point = q;
  node[r].next = -1;
  node[r].info = wt;
  (r2 < 0 ) ? (node[p].point = r) : (node[r2].next = r);
}
```

Grafos - Representação

Implemente a operação **getnode()**. Caso, julgue relevante, indique quais seriam as adaptações necessárias na estrutura proposta para representar um grafo demonstrada no slide 443.

Grafos - Representação

Implemente a operação **getnode()**. Caso, julgue relevante, indique quais seriam as adaptações necessárias na estrutura proposta para representar um grafo demonstrada no slide 443.

Em uma solução mais sofisticada podemos manter a estrutura apresentada no slide 443 e criar uma lista de nós vazios que inicialmente conterá MAXNODES. A operação getnote() apenas retirará um nó desta lista e retornará seu índice.

UNIVASF

```
void criaListaDeNodosVazios(int *listaDeNodosVazios, listaDeNodos
node)
{
  int i;
  for (i=1; i<MAXNODES; i++)
    node[i-1].next = i;
  node[i-1].next = -1;
  *listaDeNodosVazios = 0;
}

int getnode(int *listaDeNodosVazios, listaDeNodos node)
{
  int i = *listaDeNodosVazios;
  if (i!=-1)
  {
    *listaDeNodosVazios = node[*listaDeNodosVazios].next;
    return i;
  }
  printf ("\nMemoria insuficiente!\n");
  exit (1);
}
```

```
void joinwt (listaDeNodos node, int *listaDeNodosVazios, int p,
int q, int wt) {
   int r, r2;
   r2 = -1;
   r = node[p].point;
   while (r >= 0 && node[r].point != q) {
      r2 = r;
      r = node[r].next;
   }
   if (r >= 0) {
      node[r].info = wt;
      return;
   }
   r = getnode(listaDeNodosVazios, node);
   node[r].point = q;
   node[r].next = -1;
   node[r].info = wt;
   (r2 < 0 ) ? (node[p].point = r) : (node[r2].next = r);
}
```

Grafos - Representação

Implemente a operação **join()** para um grafo não ponderado de forma similar à anterior.

```
void join (listaDeNodos node, int *listaDeNodosVazios,
int p, int q)
{
   int r, r2;
   r2 = -1;
   r = node[p].point;
   while (r >= 0 && node[r].point != q) {
      r2 = r;
      r = node[r].next;
   }
   if (r >= 0) {
      return;
   }
```

Grafos - Representação

```
    r = getnode(listaDeNodosVazios, node);
    node[r].point = q;
    node[r].next = -1;
    (r2 < 0 ) ? (node[p].point = r) : (node[r2].next = r);
}
```

Grafos - Representação

Implemente a operação **remv()** que aceita ponteiros para dois nós de cabeçalho e remove o arco entre eles, caso este exista.

Observação: Considere a existência de uma função freenode().

UNIVASF