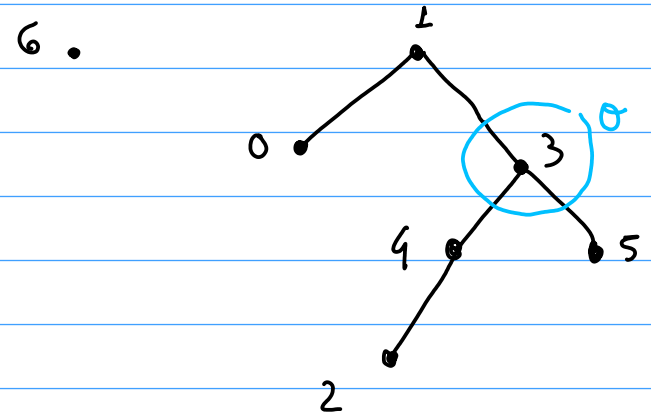
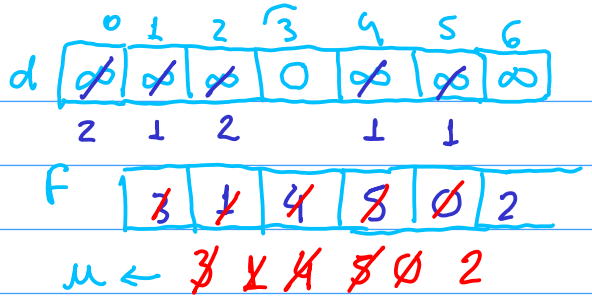
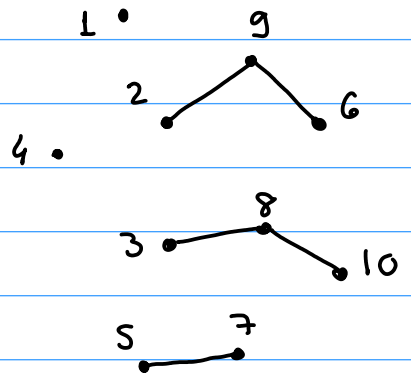




lista, fila, pilha

```
dl
format=edgelist1
n=10
data:
2 9
3 8
5 7
6 9
8 10
```



saída:

1

2 6 9

3 8 10

4

5 7

l_vért = [1, 2, 3, 4, 5 ... n]

l_arest = ["2 9 \n", "3 8 \n" ...]

1 ① print 1 /n

2 6 9 remove

3 → 8 → 10 ②

4

5 7

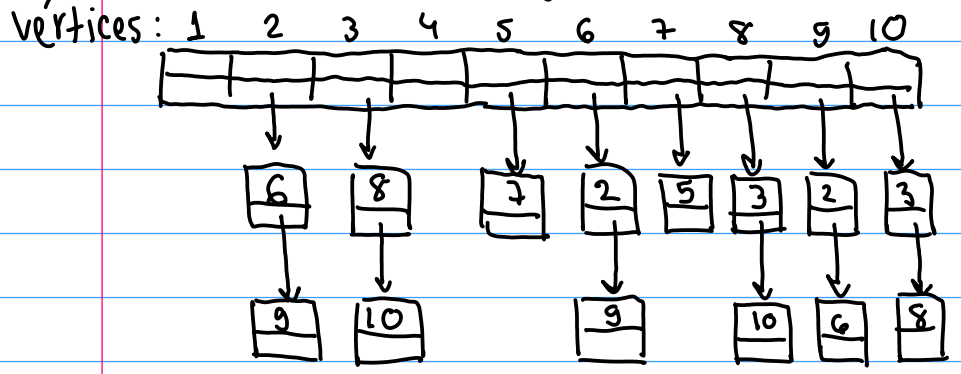
entrada:

e = ["dl /n", " format ... /n", ..., "8 10/n"]

e[2] → n (l_vért) (l_arest)

L_arestas = ["2 9/n", "3 8/n", ...]

lista de adjacências //



entrada

```
dl
format=edgelist1
n=10
data:
2 9
3 8
5 7
6 9
8 10
```

saída:

```
1
2 6 9
3 8 10
4
5 7
```

0 1 2 ... n-1

L = [[ ], [6, 9], [8, 10], [ ], [7], ..., [3, 8]]

→ criar lista L de tam. n

→ posição i da L → procurar i+1 na L_arest
e add na lista posição i de L

preencher a lista de adj.
criar lista L → L = n * [ ]
for i in range(n)
L_arest
print i
FILA: [9

L_arestas ['2 9\n', '3 8\n', '5 7\n', '6 9\n', '8 10\n']

L_a

(L_arestas[1])[0] e [2]

aux = L_a[0]

linha.append(L_a[0])

L_arestas_int [[2,9],[3,8],[5,7],[6,9],[8,10]]

i = 0, 1, 2, 3

+2 (0 1) +2

(2 3)

L_arestas_ints = [2,9,3,8,5,7,6,9,8,10]

L_aux1 =

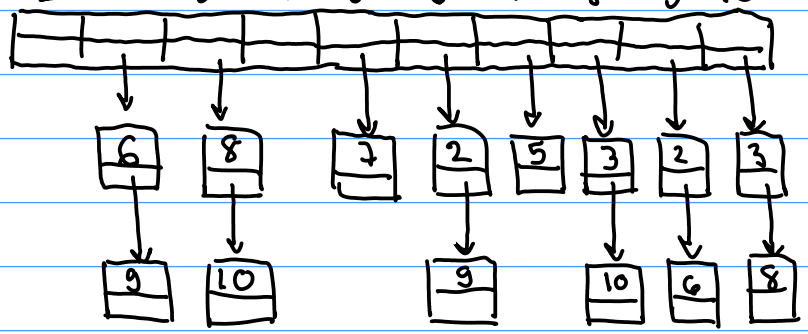
```
L_arestas_int = []*(2*len(L_arestas))
for i in range(len(L_aux1)):
  j = i
  linha_a = []*n
  linha_a.append(L_aux1[j])
  linha_a.append(L_aux1[j+1])
  j = j+2
  L_arestas_int.append(linha_a)
```

['2 9\n', '3 8\n', '5 7\n', '6 9\n', '8 10\n']

V6:

l_arestas_int : [[2,9],[3,8],[5,7],[6,9],[8,10]]

l_adj : vértices: 1 2 3 4 5 6 7 8 9 10



0 1 2 n-1

l_adj = [[ ], [6,9], [8,10], [ ], [7], ..., [3,8]]

entrada

dl
format=edgelist1
n=10
data:
2 9
3 8
5 7
6 9
8 10

saída:

1
2 6 9
3 8 10
4
5 7

vértices = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

arestas : [[2, 9], [3, 8], [5, 7], [6, 9], [8, 10]]

[[2, 5], [1, 3], [2], [ ], [1]]

lista : [[ ], [9], [8], [ ], [7], [9], [5], [3, 10], [2, 6], [8]]

adj 0 1 2 3 4 5 6 7 8 9

1+1

(n-1)

visitados : [F, F, F, ... F]

n vezes

componentes = [ ]

verificar visitados[p] = V

componente = [1, 2, 3, 5]

VISITAR // criar componente [ ]

vizinhos = lista_adj[i] → visitados[i] = V → componente.append(i+1)

vizinhos[j] → visitados

entrada

```
dl
format=edgelist1
n=10
data:
2 9
3 8
5 7
6 9
8 10
```

saída:

```
1
2 6 9
3 8 10
4
5 7
```