

- Desenvolva um programa que faça uso das função básicas de manipulação de uma **pilha**.
Desenvolva sua própria classe Pilha. Utilize Generic.
Implemente a pilha em **Vetor**.
- Desenvolva um programa que faça uso das função básicas de manipulação de uma **fila circular**.
Desenvolva sua própria classe Fila. Utilize Generic.
Implemente a fila em **Vetor**.
- Desenvolva um programa e implemente as função básicas de manipulação de uma **lista simplesmente encadeada**.
Desenvolva sua própria classe ListaSimples. Utilize Generic.
- Desenvolva um programa e implemente as função básicas de manipulação de uma **lista duplamente encadeada**.
Desenvolva sua própria classe ListaDupla. Utilize Generic.
- Desenvolva um programa que conte a **frequência de cada palavra de um texto**. Leia esse texto de um arquivo.
Utilize uma lista simplesmente encadeada. Este programa deve permitir salvar o resultado em um arquivo binário e texto e ler este arquivo em binário e texto.
- Implemente a **Pilha** da questão 1 em uma estrutura de **Lista Encadeada**.
- Implemente a **Fila** da questão 2 em uma estrutura de **Lista Encadeada**.
- Desenvolva um programa que implemente o algoritmo que **leia uma expressão já na forma pós-fixa e a resolva**.
Exemplo: Expressão infixa: $304 + 11 - 2$ Expressão pós-fixa: $304 11 + 2 -$ Resultado: 313

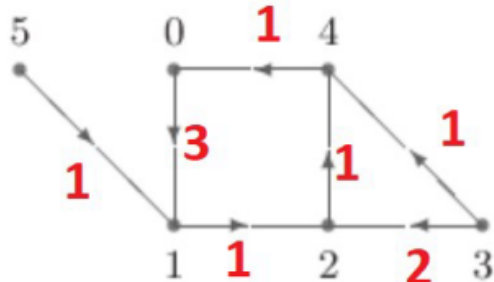
Algoritmo:

- Inicialize pilha vazia
- Varra a expressão:
 - Se for operando, **empilhe** o valor.
 - Se for operador, **desempilhe** dois valores da pilha, efetue a operação com eles e **empilhe** o resultado.
- Ao final, exiba o elemento no topo da pilha, que é o resultado da expressão.

- Desenvolva um programa que conte as distâncias de uma cidade para todas as outras cidades.

Algoritmo:

- Inicialize o vetor resultado com -1, exceto o elemento da cidade inicial, que deve ser zero.
- Insira na Fila a cidade inicial.
- Enquanto Fila não for vazia:
 - Remover cidade da Fila, que será a cidade atual
 - Percorra todas as cidades (cidade visitada):
 - Descubra distancia (aresta) entre a cidade atual e a cidade visitada
 - Se, a aresta for maior que zero e o elemento resultado da cidade visitada ainda é -1: a distancia com a cidade visitada é igual a distancia da cidade atual mais a aresta.
 - Inserir cidade visitada na Fila (que no próximo loop será a cidade atual).

Exemplo	Entradas	Saída																																																													
 <p>OBS: números em vermelho são as distâncias.</p>	<p>cidade inicial = 3</p> <table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr><tr><td>0</td><td>0</td><td>3</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>2</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>3</td><td>0</td><td>0</td><td>2</td><td>0</td><td>1</td><td>0</td></tr><tr><td>4</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>5</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>		0	1	2	3	4	5	0	0	3	0	0	0	0	1	0	0	1	0	0	0	2	0	0	0	0	1	0	3	0	0	2	0	1	0	4	1	0	0	0	0	0	5	0	1	0	0	0	0	<p>Vetor com as distâncias</p> <table><tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr><tr><td>2</td><td>5</td><td>2</td><td>0</td><td>1</td><td>-1</td></tr></table> <p>OBS: representar com -1 cidades não alcançadas.</p>	0	1	2	3	4	5	2	5	2	0	1	-1
	0	1	2	3	4	5																																																									
0	0	3	0	0	0	0																																																									
1	0	0	1	0	0	0																																																									
2	0	0	0	0	1	0																																																									
3	0	0	2	0	1	0																																																									
4	1	0	0	0	0	0																																																									
5	0	1	0	0	0	0																																																									
0	1	2	3	4	5																																																										
2	5	2	0	1	-1																																																										