

Questão 1.

Acrescente à definição do TAD Lista definido no arquivo `List.h` as seguintes funções:

1. `List *List_invert(List *list):`
a função deve receber uma lista como parâmetro e devolver uma nova lista na ordem inversa.
2. `int List_isIn(List *list, void *value, int (*compar)(void*,void*)):`
a função deve receber uma lista e um valor e verificar se o valor pertence a algum nó da lista retornando 1 em caso afirmativo e 0 caso o valor não se encontre na lista.

Questão 2.

Escreva uma função recursiva para contar o número de elementos em uma lista ligada.

Questão 3.

Crie um programa que receba o nome de um arquivo `.txt` contendo nomes de empresas e URL's gravados um par por linha, separados por um espaço simples. Exemplo:

```
IBM https://ibm.com/augue.jpg
Bluehost http://bluehost.com/leo.jsp
Multiply https://multiply.com/morbi/odio/odio.json
Google http://google.com.au/porta/volutpat.png
```

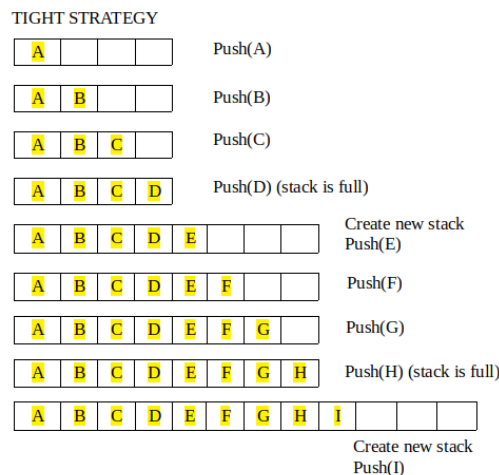
Armazene esses dados em uma lista encadeada. Escreva a respectiva estrutura e uma função que, dado o nome de uma empresa, busque o seu link correspondente na lista e ao mesmo tempo mova o nó que contém o nome buscado para o início da lista, de forma que ele possa ser encontrado mais rapidamente na próxima vez que for buscado.

Questão 4.

Uma pilha baseada em array expansível pode ser implementada alocando uma nova memória maior do que a memória da pilha anterior e copiando elementos da pilha antiga para a nova. E então, finalmente, mude o nome da nova pilha para o nome que foi dado à pilha antiga.

Existem duas estratégias para a pilha crescente:

1. **Estratégia de crescimento fixo:** adicione uma quantidade constante à pilha antiga ($N + c$)
2. **Estratégia de crescimento variável:** dobre o tamanho da pilha antiga ($2N$)



Existem duas operações nesta implementação de pilha:

1. **Operação Push Regular:** adicione um elemento no topo da pilha,
2. **Operação Push Especial:** crie uma nova pilha de tamanho maior do que a pilha antiga (de acordo com uma das estratégias acima) e copie todos os elementos da pilha antiga e, em seguida, empurre o novo elemento para a nova pilha.

Questão 5.

Considere o problema de um rato que tenta encontrar o caminho para uma saída em um labirinto (Figura 1). O rato espera escapar do labirinto tentando sistematicamente todas as rotas. Se chegar a um beco sem saída, ele refaz seus passos até a última posição com pelo menos mais um caminho não experimentado. Para cada posição, o rato pode ir em uma das quatro direções: direita, esquerda, baixo, cima. Independentemente de quão próximo esteja da saída, ele sempre tenta os caminhos inexplorados em ordem aleatória, o que pode levar a alguns desvios desnecessários. Ao reter informações que permitem retomar a busca depois que um beco sem saída é alcançado, o rato usa um método chamado *backtracking*.

O labirinto na Figura 1 é representado como uma matriz em que as passagens são marcadas com 0s, paredes com 1s e a posição de saída pelo número 2.

Implemente o jogo descrito acima e utilize uma pilha de posições na matriz como memória, para que o rato guarde as informações necessárias à escapada.

- Inicie o rato em uma posição aleatória válida,
- Selecione aleatoriamente a próxima direção que o rato vai seguir (um número de 0 a 3).
- Certifique-se de que a próxima posição é válida e sinalize se ele escapou ou não.
- Crie um loop de jogo para que o rato continue procurando até encontrar uma saída.


	0	1	2	3	4	5	6	7	8	9	10
0	1	1	1	1	1	1	1	1	1	1	1
1	1	0	0	0	1	0	0	0	0	0	1
2	1	0	1	0	1	1	1	0	1	0	1
3	2	0	1	0	0	0	0	0	1	0	1
4	1	0	1	1	1	1	1	0	1	0	1
5	1	0	1	0	1	0	0		1	0	1
6	1	0	0	0	1	0	1	0	1	0	1
7	1	1	1	1	1	0	1	0	0	0	1
8	1	0	1	0	1	0	1	0	1	1	1
9	1	0	0	0	0	0	1	0	0	0	1
10	1	1	1	1	1	1	1	1	1	1	1

Figura 1: Matriz de números representando um labirinto.

- Toda vez que uma direção que contém uma posição válida é encontrada o rato segue nessa direção e empilha a nova posição em que se encontra, se o caminho resultar em um beco sem saída você deve desempilhar as direções e ir testando até encontrar uma posição que possua uma sucessora ainda não visitada.

Questão 6.

Escreva um programa simples de reserva de passagens aéreas. O programa deve apresentar um menu com as seguintes opções: reservar passagem, cancelar reserva, verificar se a passagem está reservada para determinada pessoa e exibir os passageiros.

As informações são mantidas em uma lista encadeada em ordem alfabética de nomes. Suponha que as passagens sejam reservadas para apenas um voo. Após a reserva exiba a lista de passageiros do voo.

Questão 7.

Escreva um programa que simule o controle de uma pista de decolagem de aviões em um aeroporto. Neste programa, o usuário deve ser capaz de realizar as seguintes tarefas:

- Listar o número de aviões aguardando na pista de decolagem;
- Autorizar a decolagem do primeiro avião da fila;
- Adicionar um avião à fila de espera;
- Listar todos os aviões na fila de espera;
- Listar as características do primeiro avião da fila.

Considere que os aviões possuem um nome e um número inteiro como identificador. Adicione outras características conforme achar necessário.

Questão 8.

Escreva uma função que percorre uma lista ligada de inteiros e remove (liberando adequadamente a memória) todos os nós que contêm elementos de valor zero como mostra a Figura 2.

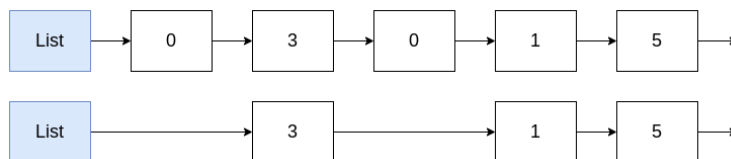


Figura 2: Remoção de zeros da lista.

Questão 9.

Seu Dionízio abriu um restaurante e precisa organizar os pedidos que chegam na cozinha para que os clientes não saiam insatisfeitos com o atendimento. Ele então resolveu recorrer a um sistema que permita a ele gerenciar a ordem dos pedidos que entram e saem da cozinha, além de calcular o valor dos pedidos das mesas para que nenhum garçom erre as contas.

- Crie uma estrutura `Prato` com os campos: nome e valor. Crie e inicialize um vetor da estrutura `Prato` com alguns pratos do restaurante.
- Crie uma estrutura `Pedido` com os membros: número do pedido e uma lista de pratos.
- Implemente uma função para criar um pedido. Os pratos selecionados devem ser inseridos na lista de pratos do pedido.
- Os pedidos devem ser colocados em uma fila para simular o seu preparo pela cozinha do restaurante.
- Crie uma função que retira um pedido da fila de preparo da cozinha e imprime os pratos do pedido, soma os preços dos pratos e apresenta o total do pedido.
- Crie uma função para cancelar um prato específico de um pedido.