Lista de Atividades 04

➔ **Todos os exercícios a seguir deverão ser implementados em um único arquivo.**

➔ **Construa um menu de opções onde o usuário possa escolher o exercício que deseja executar, assegure-se de que a execução tenha uma pausa ao final de cada exercício, e retorne ao menu inicial após isso.**

➔ **Construa uma função *menu*, que será chamada dentro da função *main*, esta função não deve receber nenhum parâmetro como argumento. Dentro desta função deve ser executada uma limpeza da tela, e a exibição das opções disponíveis, como por exemplo:**

1 - Exercício_01

2 - Exercício_02

... etc ...

O tratamento da opção escolhida pelo usuário deverá ser feito dentro da função *menu* usando um *switch case*, em que cada opção executará apenas a chamada da função correspondente ao exercício a ser realizado.

Utilize *do while* para assegurar o retorno à exibição do menu de opções após a execução de qualquer exercício. E utilize 0 (zero) como opção de saída.

➔ **Na função *main*, deve haver apenas a chamada para a função *menu*.**

➔ **Construa quantas funções julgar necessário! Lembrando que quanto mais genérica for uma função, mais fácil será o reaproveitamento de código.**

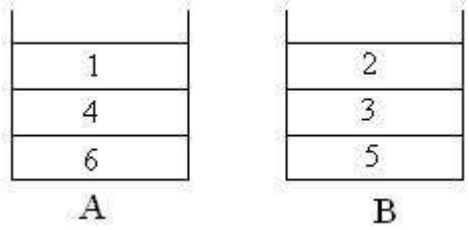
➔ **Utilize a seguinte *struct* para construção dos nós das listas:**

```
typedef struct no {
  int info;              // campo de imformação
  struct no *prox;       // ponteiro para o proximo nó d alista
} NO;
```

Exercícios

Os exercícios a seguir são iguais aos da Lista 02, no entanto, desta vez, realize-os utilizando LISTAS SIMPLESMENTE ENCADEADAS.

1. Crie duas PILHAS e as alimente conforme as PILHAS da figura abaixo, observando que em A e B os itens do topo são 1 e 2, respectivamente. Crie uma terceira PILHA onde estarão organizados os números em ordem crescente, de baixo para cima. A passagem dos valores das PILHA A e B para a terceira PILHA deve passar por uma estrutura de comparação de dados, para checar qual dado de qual PILHA deverá entrar na terceira PILHA em dado momento com vistas a assegurar a ordenação.



2. Implementar a simulação computacional de uma FILA de banco, considerando as seguintes condições/restrições:

a) FILA com comprimento máximo de 20 clientes. (Em uma situação real, esta seria a condição para abertura de mais um caixa de atendimento, tal situação não será tratada neste exercício)

b) Prioridade de atendimento para clientes maiores de 60 anos.

c) A cada operação de inserção e/ou remoção de cliente na FILA, a FILA deve ser exibida na tela.

d) Observar tratamento de erros e RESTRIÇÕES associados a FILAS.

e) no vetor de dados da struct FILA, guarde as idades dos clientes presentes na fila.

➔ Dica brutal!!!! Pode usar duas filas de clientes!!!