

## Aula 4 - Lista 4.2 de exercícios de Programação e Estruturas de Dados II

### Problema 1

Uma pilha de inteiros é descrita pelas seguintes declarações:

```
#define MAX 100
```

<code>int vet[MAX];</code>	<code>/* vetor com os elementos da pilha */</code>
<code>int num;</code>	<code>/* número de elementos na pilha */</code>
<code>void init(void);</code>	<code>/* inicia a pilha */</code>
<code>void push(int elem);</code>	<code>/* empilha elem */</code>
<code>int pop(void);</code>	<code>/* desempilha um elemento, que e' o resultado */</code>
<code>int vazia(void);</code>	<code>/* testa se pilha esta' vazia */</code>
<code>void show(void);</code>	<code>/* imprime os elementos da pilha, entre colchetes, em uma única linha, em ordem, a partir do topo */</code>

Terminada a implementação, o seguinte programa é executado:

```
int main(void) {
    int a,b;
    init();
    show();
    push(3); push(4); push(5); push(6);
    show();
    a=pop(); b=pop(); push(a+b);
    show();
    a=pop(); push(a*pop());
    show();
    printf("%d", pop());
    show();
    return 0;
}
```

Mostre qual será o efeito deste programa, indicando qual será a saída impressa pelo programa.

Comando	Conteúdo da pilha: num, vet	Saída impressa
<code>init();</code>	0, [ ? ? ? ? ? ... ? ]	
<code>show();</code>		[ ]
<code>push(3);</code>	1, [ 3 ? ? ? ? ... ? ]	
<code>push(4);</code>	2, [ 3 4 ? ? ? ... ? ]	
<code>push(5);</code>	3, [ 3 4 5 ? ? ... ? ]	
<code>push(6);</code>	4, [ 3 4 5 6 ? ... ? ]	
<code>show();</code>		[ 6 5 4 3 ]
<code>a=pop();</code>	3, [ 3 4 5 6 ? ... ? ] (a=6)	
<code>b=pop();</code>	2, [ 3 4 5 6 ? ... ? ] (b=5)	
<code>push(a+b);</code>	3, [ 3 4 11 6 ? ... ? ]	
<code>show();</code>		[ 11 4 3 ]
<code>a=pop();</code>	2, [ 3 4 11 6 ? ... ? ] (a=11)	
<code>push(a*pop());</code>	2, [ 3 44 11 6 ? ... ? ] (pop tem resultado 4)	
<code>show();</code>		[ 44 3 ]
<code>printf("%d", pop());</code>	1, [ 3 44 11 6 ? ... ? ] (pop tem resultado 44)	44
<code>show();</code>		[ 3 ]

### Problema 2

Escreva uma função que receba duas Listas (L1 e L2), intercale-as gerando uma terceira Lista, L3. Faça versões estáticas e dinâmicas do código.

**Problema 3**

Escreva uma função que inverte L1, colocando o resultado em L2. Faça versões estáticas e dinâmicas do código.

**Problema 4**

Considere uma pilha P vazia e uma fila F não vazia. Utilizando apenas os testes de fila e pilha vazias, as operações Enfileira, Desenfileira, Empilha, Desempilha, e uma variável aux do TipoItem, escreva uma função que inverta a ordem dos elementos da fila.

**Problema 5**

Para um dado número inteiro  $n > 1$ , o menor inteiro  $d > 1$  que divide  $n$  é chamado de fator primo. É possível determinar a fatoração prima de  $n$  achando-se o fator primo de substituindo  $n$  pelo quociente  $n / d$ , repetindo essa operação até que  $n$  seja igual a 1. Utilizando um dos TADs vistos em sala (Lista, Pilha ou Fila) para auxiliá-lo na manipulação de dados, implemente uma função que compute a fatoração prima de um número imprimindo os seus fatores em ordem decrescente. Por exemplo, para  $n=3960$ , deverá ser impresso  $11 * 5 * 3 * 3 * 2 * 2 * 2$ . Justifique a escolha do TAD utilizado.

**Problema 6**

Considere a implementação de filas usando arranjos “circulares”. Escreva uma função FuraFila(TipoFila\* pFila, TipoItem x) que insere um item na primeira posição da fila. O detalhe é que seu procedimento deve ser  $O(1)$ , ou seja, não pode movimentar os outros itens da fila. (observe que este neste caso, estaremos desrepeitando o conceito de FILA – primeiro a entrar é o primeiro a sair).

**Problema 7**

Como você implementaria uma fila de pilhas? Uma pilha de filas? Uma fila de filas? Escreva rotinas para implementar as operações corretas para cada uma destas estruturas de dados.

**Problema 8**

Implemente uma fila em C, onde cada item da fila consista em um número variável de inteiros.