ORIENTAÇÕES PARA A ATIVIDADE 1

DESENVOLVIMENTO DO TRABALHO:

1) Considere o programa abaixo e depois dê o que se pede:

```
#include
<iostream>
using
namespace std;
void
Teste1(int ); void
Teste2(int &); int
Teste3(int);
int x = 20;

int main(void)  {
        int numero = 10,
        outroNumero;
        Teste1(numero);
        cout << "Valor de numero (após Teste1) = " <<
        numero << endl; cout << "X = " << x << endl;
        Teste2(numero);
        cout << "Valor de numero (após Teste2) = " <<
        numero << endl; cout << "X = " << x << endl;
        outroNumero = Teste3(numero);
        cout << "Valor de outro numero (após Teste3) = " <<
        outroNumero << endl; cout << "X = " << x << endl;
        }

void Teste1(int numero) {
        numero =
        numero +x ; x+
        +;
        }

void Teste2(int &numero) {
        int valor = 100;
        numero = numero +
        valor; x++;
        }

int Teste3(int n) {
        int valor =
        200; n = n
        + valor;
        x--;
        return n;
        }
```

Pede-se:

a) Identifique as variáveis globais e locais. Quando identificar uma variável local, especifique o escopo da mesma.
b) Identifique, em cada função, o tipo de passagem de parâmetros.
c) Mostre, passo a passo, o valor de todas as variáveis, indicando o momento em que as variáveis não mais ocupam espaço na memória.
d) Diga o que é impresso na tela

2) Considere listas lineares sequenciais não ordenadas de inteiros não nulos. Faça um programa para:

a) Construir duas listas sem repetição de dados. Para isto, implemente uma função de nome inserirSemRepetir que receba como parâmetros: o vetor de dados, o elemento a ser inserido, a quantidade de elementos no vetor e o tamanho máximo definido para o vetor.
Note:

- Deverão ser emitidas mensagens de erro adequadamente.
- Será preciso fazer uma busca sequencial para evitar repetição de dados.
- A função deverá ser chamada repetidamente para criar cada uma das listas

b) Imprimir as listas criadas no item **a**, implementando uma função de nome **listar** (ou percorrer), conforme estudado em aula.
c) Intercalar as listas criadas, gerando uma terceira lista sequencial. Por exemplo, a 1ª. lista possui os elementos 10, 34 e 5 e a 2ª. lista possui os elementos 4, 7 e 9. A lista resultante será 10, 4,34,7,5 e 9.
d) Gerar uma lista que seja a interseção das listas do item **a**, como em interseção de conjuntos.
e) Imprimir as listas geradas nos itens **c** e **d**, usando a função listar (ou percorrer).
f) Gerar uma lista que seja a união das listas do item **a** e depois imprimi-la.
g) Remover um elemento da lista gerada no item **f** através do índice passado. Para isto, implemente uma função com o seguinte protótipo :

void removerPeloIndice(int [], int, int);

Parâmetros :

- vetor de elementos
- quantidade de elementos no vetor
- índice do valor a ser removido

Após a leitura do índice, verifique sua validade. Caso não seja válido, emita mensagem de erro na **main**, caso contrário chame a função para realizar a remoção.