ESTRUTURA DE DADOS – P1

1) Considere duas Listas L1 e L2 já criadas e com os elementos inseridos ordenados. Escreva a função intercala, que retorna uma terceira lista, também ordenada, criada a partir da intercalação entre os elementos das listas L1 e L2. Não é para inserir todo mundo e depois ordenar. É para intercalar as listas. As estruturas de dados e o protótipo da função:

```
struct Lista{
    ElementoLista* prim;
};

struct ElementoLista{
    int chave;
    ElementoLista* prox;
};

Lista* intercala(Lista* l1, Lista *l2);
```

2) Uma lista encadeada tem vários elementos repetidos. A estrutura de dados desta lista é a mesma da questão 1. Você deve criar uma nova lista. Cada elemento desta nova lista terá um campo adicional, com o número total de elementos encontrados com aquela chave.
    a) Defina a nova estrutura de dados.
    b) Escreva o protótipo da função copiaEliminandoRepetidos levando em consideração as estruturas de dados pertinentes.
    c) Escreva a função, supondo que a lista esteja inicialmente ordenada.
    d) Qual é a complexidade do seu algoritmo? Justifique.
    e) Se a lista não tivesse ordenada, a complexidade seria a mesma? Explique.

3) O que faz a função recursiva abaixo?

```
int f(int a, int b){
    if(b == 1) return a;
    else return a + f(a, b-1);
}

int main(){
    cout << f(5,1000);
    return 0;
}
```

4) Sobre o quick sort:
    a) Considere a sequência 10 8 4 3 1 5 2 7. Particione a sequência usando o último elemento como pivô. Mostre passo a passo a troca de elementos.
    b) Este é o melhor pivô? Justifique.
    c) Este é o pior pivô? Justifique.
    d) Se em todos os particionamentos o melhor pivô for escolhido, qual será a complexidade do algoritmo de ordenação.
    e) E se o pior pivô for escolhido, qual será a complexidade?

5) O programa abaixo computa a sequência de Fibonacci. Além da sequência, uma variável contador foi acrescentada para contar quantas chamadas recursivas são realizadas para cada caso. Diga o que será impresso na tela no final da execução do seu programa. Explique como você chegou a este resultado.

```
int contador = 0;

int fib(int n){
    contador++;
    if(n == 0) return 0;
    if(n == 1) return 1;
    return fib(n-2) + fib(n-1);
}

int main(){
    contador = 0;
    cout << fib(4) << " ";
    cout << contador << endl;
    contador = 0;
    cout << fib(5) << " ";
    cout << contador << endl;

    return 0;
}
```