Universidade do Estado de Santa Catarina
Ciência da Computação
Complexidade de Algoritmos
Exercício Prático I

Dada a estrutura de dados:

```
struct Agenda
{
        int cod;
        char nome[30];
        char end[30];
        char fone[20];
}
```

1) Implemente a intercalação (*merge*) entre dois arquivos binários de dados do tipo *struct Agenda*, considere que os arquivos estão previamente ordenados. A intercalação de deve ser feita em disco, não é permitido que todo conteúdo dos arquivos esteja armazenado na memória em um determinado instante. O resultado deve ser armazenado em um terceiro arquivo.
2) Implemente funções para inserção, remoção e consulta de dados do tipo *struct Agenda* em uma lista encadeada. A chave para pesquisa deve ser o código. Calcule a complexidade de tempo e espaço de cada uma das funções. Deve ser implementada uma interface para entrada de dados, para que possam ser executados os testes.
3) Implemente funções para inserção, remoção, consulta e impressão em ordem de dados do tipo *struct Agenda* em uma árvore binária de pesquisa. A chave para pesquisa deve ser o código. As funções não devem ser implementadas usando recursão. Calcule a complexidade de tempo e espaço de cada uma das funções. Deve ser implementada uma interface para entrada de dados, para que possam ser executados os testes.
4) Faça a medição dos tempos de execução dos métodos de ordenação: *bubble sort*, *insertion sort, merge sort*, *quick sort e heap sort*. Use vetores de inteiros para os teste, apresente os tempos para o melhor e pior caso de cada método, apresente os resultados na forma de um gráfico, onde o eixo *x* representa o número de elementos do vetor e o eixo *y* o tempo de execução.
5) Implemente o algoritmo de Dijkstra que calcula o caminho mais curto entre dois vértices de um grafo. Mostre a complexidade de sua implementação.