Miniteste 2 – Exemplo 2 (adaptado de miniteste 2012/13)

1 Tendo por base as bibliotecas de estruturas de dados apresentadas em Programação 2, implemente as funcionalidades pedidas nas três alíneas seguintes no ficheiro **prob1.c**. Sempre que conveniente utilize as funções disponíveis.

1.1 Implemente a função `vetor_pilha` que cria uma nova pilha a partir dos elementos guardados num vetor. Quando retirados da pilha, os elementos devem estar por ordem alfabética.

```
pilha* vetor_pilha(vetor *v)
```

O parâmetro da função é o apontador para o vetor original. A função deve devolver a nova pilha se for bem sucedida e NULL se ocorrer algum erro. O vetor original não deve ser alterado.

Depois de implementada a função, o programa deverá apresentar:

```
Vetor: Estocolmo Oslo Helsinquia Copenhaga Reykjavik
Pilha: Copenhaga Estocolmo Helsinquia Oslo Reykjavik
```

1.2 Implemente a função `tabela_redimensiona` que altera o tamanho de uma tabela de dispersão.

```
int tabela_redimensiona(tabela_dispersao *td, int novo_tamanho)
```

O primeiro parâmetro da função é o apontador para a tabela de dispersão e o segundo parâmetro é o novo tamanho da tabela. A função deve devolver 1 se for bem sucedida ou 0 se ocorrer algum erro.

Depois de implementada a função, o programa deverá apresentar:

```
Tabela de dispersao redimensionada corretamente.
```

1.3 Implemente a função `descobre_segredo` que retorna uma lista com as palavras que compõem o segredo contido numa árvore AVL. Cada nó da árvore contém uma palavra.

```
lista* descobre_segredo(avl_arvore *avl, lista *indicacoes)
```

Os parâmetros da função são o apontador para a árvore AVL e para a lista contendo os caminhos a seguir na árvore. A lista `indicacoes` contém apenas as palavras “esquerda” e “direita”, que indicam os ramos da árvore a percorrer a partir da raiz, que contém a primeira palavra. A árvore original não deve ser alterada.

Depois de implementada a função, o programa deverá apresentar:

```
Segredo: prog2 mt2 e' facil
```

Extra: implemente uma versão recursiva da função `descobre_segredo` que imprime (neste caso não retorna uma lista) as palavras que compõem o segredo.

```
void descobre_segredo_recurs(no_avl *no, lista *indicacoes)
```

2 Utilizando a estrutura de dados *min-heap*, pretende-se saber os 5 melhores tempos de execução dos trabalhos submetidos pelos estudantes de Prog2. A lista de tempos é guardada num ficheiro com o seguinte formato: `grupo minutos segundos`. O nome do grupo é composto por apenas 1 palavra (máximo 20 carateres). Implemente no ficheiro **prob2.c** a seguinte função:

```
int topSubmissoes(FILE *f)
```

O parâmetro da função é o apontador para o ficheiro. A função deve devolver 1 no caso de ser bem sucedida e 0 em caso contrário. Considere que o ficheiro tem, no máximo, 100 grupos.

Depois de implementada a função, o programa deverá apresentar:

```
Top 5 submissoes:
Grupo G095 - 0min 5seg
Grupo G011 - 0min 10seg
Grupo G096 - 0min 14seg
Grupo G052 - 0min 55seg
Grupo G036 - 1min 10seg
```