Miniteste 2 – Exemplo

1 Tendo por base a biblioteca de filas e pilhas, implemente as funcionalidades pedidas nas duas alíneas em baixo. A biblioteca encontra-se disponível nos ficheiros `pilha.h`, `pilha.c`, `fila.h` e `fila.c`. Sempre que achar conveniente utilize as funções já disponíveis.

1.1 Acrescente uma nova funcionalidade à biblioteca pilha, que crie uma cópia de uma pilha, retornando um apontador para a pilha criada:

```
pilha* pilha_duplica(pilha *p)
```

O parâmetro da função é o apontador para a pilha original. A função deve devolver uma pilha no caso de ser bem sucedida ou NULL caso não tenha sido possível copiar a pilha.

Depois de implementada a função, o programa deverá apresentar:

```
EXERCICIO 1.1:
Resultado obtido e' o esperado.
```

1.2 Acrescente uma nova funcionalidade à biblioteca fila remove todos os elementos começados por um certo caracter:

```
int fila_elimina(fila *f, char c_inicial)
```

Esta função deverá retornar o número de elementos que foram removidos da fila original.

Depois de implementada a função, o programa deverá apresentar:

```
EXERCICIO 1.2:
Resultado obtido e' o esperado.
```

**** Utilize o ficheiro prob1.c ****

2 Tendo por base a biblioteca de tabelas de dispersão que desenvolveu no trabalho prático 2, implemente as funcionalidades pedidas nas duas alíneas em baixo. A biblioteca encontra-se disponível nos ficheiros `tabdispersao.h` e `tabdispersao.c`. Sempre que achar conveniente utilize as funções já disponíveis.

2.1 Acrescente uma nova função à biblioteca, que remova os vários elementos de uma tabela de dispersão:

```
int tabela_remove_varios(tabela_dispersao *td, vetor *elementos)
```

O primeiro parâmetro da função é o apontador para a tabela de dispersão e o segundo parâmetro é o apontador para um vector com os elementos a remover. A função deve ignorar os elementos do vetor que não existam na tabela. A função deve ainda devolver 1 no caso de ser bem sucedida e 0 em caso contrário.

Depois de implementada a função, o programa deverá apresentar:

```
Num elementos antes de remover: 45
Num elementos depois de remover: 42
```

2.2 Acrescente uma nova função à biblioteca que consiste no retorno de alguma informação sobre a tabela:

```
int tabela_info(tabela_dispersao *td, float *factorCarga, int *maxElems)
```

O primeiro parâmetro da função é o apontador para a tabela de dispersão e os dois restantes são utilizados para retornar 1) o fator de carga da tabela e 2) o número máximo de elementos associados a uma posição. A função deve devolver 1 no caso de ser bem sucedida e 0 em caso contrário.

Depois de implementada a função, o programa deverá apresentar:

```
Fator de carga: 3.46
Numero max de elementos: 6
```

**** Utilize o ficheiro prob2.c ****

3 Considere a estrutura de dados min-heap implementada nos ficheiros `heap.h`, e `heap.c`. Utilizando essa estrutura, pretende-se imprimir por ordem crescente os valores de leituras guardadas num ficheiro de texto:

```
int imprimirLeiturasOrdenadas (const char *nomeFicheiro)
```

O parâmetro da função é o nome do ficheiro. A função deve devolver 1 no caso de ser bem sucedida e 0 em caso contrário.

Depois de implementada a função, o programa deverá apresentar:

```
511
621
1576
(...)
98954
99058
99874
```

**** Utilize o ficheiro prob3.c ****