Pesquisa e Classificação de Dados

2023/2 - **Trabalho da disciplina**

Enunciado

O trabalho consiste na implementação, em C, de um sistema de cadastros. Considere que existe um arquivo chamado `banco.txt` que contém uma coleção de cadastros de usuários. A primeira linha do arquivo contém um inteiro indicando a quantidade de usuários cadastrados. Cada uma das próximas linhas contém um cadastro, no formato `[ID] [Nome Completo] [Idade]`. O campo `[ID]` é uma string aleatória (única para cada cadastro) com letras minúsculas; O campo `[Nome Completo]` é uma string com maiúsculas, minúsculas e espaços; o campo `[Idade]` é um inteiro.

Como exemplo, considere o arquivo `banco.txt` com o seguinte conteúdo:

```
5
afd James Paul McCartney 81
dwiz Richard Starkey 83
afdg John Winston Ono Lennon 40
dafd Larissa de Macedo Machado 30
dwiop George Harrison 58
```

Neste arquivo há o cadastro de 5 usuários: o primeiro tem ID `afd`, nome `James Paul McCartney` e idade 81; o segundo tem ID `dwiz`, nome `Richard Starkey` e idade 83; e assim por diante.

Escreva um programa em C que lê o arquivo `banco.txt` e o carrega para a estrutura de dados definida na próxima seção. Em seguida, seu programa deve ser uma sequência de comandos do usuário, onde cada comando pode ser:

- `? [ID]` : busca na estrutura de dados o cadastro com dado ID. Caso não seja encontrado, imprima `ID [ID] nao encontrado.`. Caso contrário, imprima o cadastro no formato `(ID|Nome Completo|Idade)` (veja exemplo abaixo);
- `+ [ID] [Nome Completo] [Idade]` : insere um novo cadastro na estrutura de dados. Você pode assumir que o usuário não irá inserir um `[ID]` que já está na estrutura. Após a inserção, imprima `Inserido ([ID]|[Nome Completo]|[Idade])` (veja exemplo abaixo);
- `- [ID]` : remove da estrutura de dados o cadastro com dado ID. Se o ID dado não está na estrutura, imprima `ID [ID] nao encontrado.`. Caso contrário, após a remoção, imprima `Removido ([ID]|[Nome Completo]|[Idade])` (veja exemplo abaixo);
- `P` : imprime todos os cadastros na estrutura de dados, um por linha, no formato `([ID]|[Nome Completo]|[Idade])`, em qualquer ordem (veja exemplo abaixo);
- `S` : salva todos os cadastros que estão na estrutura de dados no arquivo `banco.txt`, no formato dado acima. Ao ler este comando, o arquivo `banco.txt` deve ser reconstruído "do zero", com (e apenas com) os cadastros que estão na estrutura de dados no momento em que o comando é lido. Após salvar, imprima `banco.txt salvo.`.
- `F` : termina a execução.

Confira o seguinte exemplo de execução (o caracter `>` indica entrada do usuário; considere que o arquivo `banco.txt` contém, inicialmente, os 5 registros dados como exemplo acima):

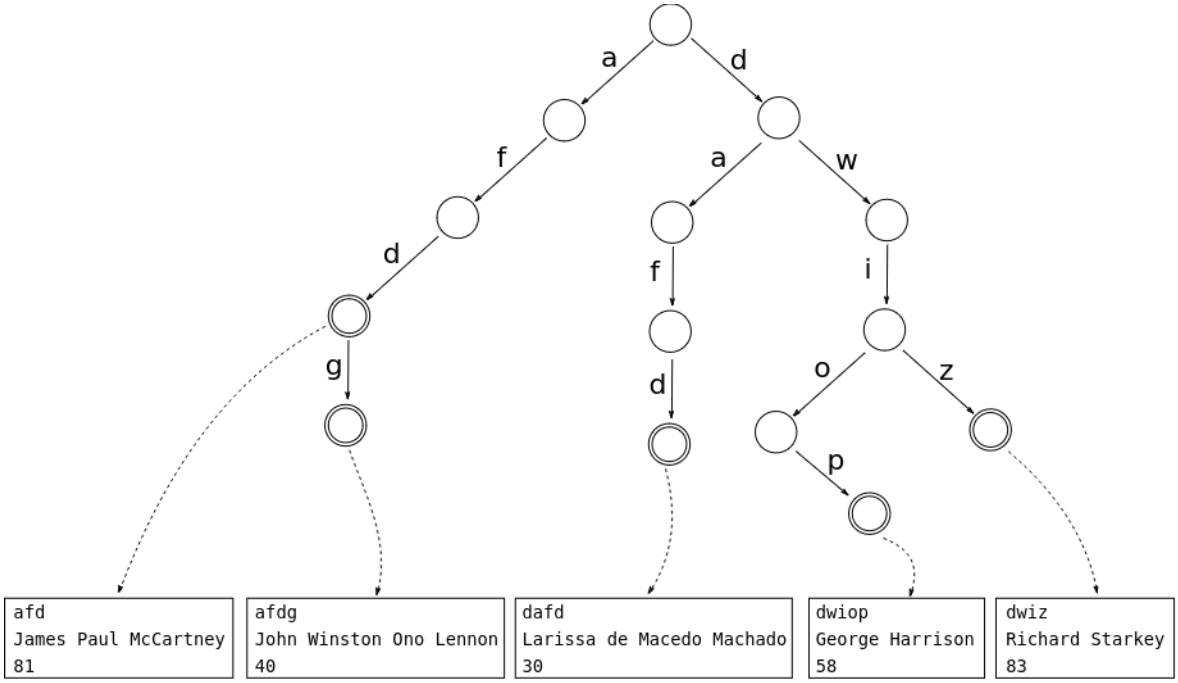
```
> P
(afd|James Paul McCartney|81)
(afdg|John Winston Ono Lennon|40)
(dafd|Larissa de Macedo Machado|30)
(dwiop|George Harrison|58)
(dwiz|Richard Starkey|83)

> ? afd
(afd|James Paul McCartney|81)

> ? afdg
(afdg|John Winston Ono Lennon|40)

> ? af
ID af nao encontrado.

> ? rusbe
ID rusbe nao encontrado.

> + rusbe Michael Joseph Jackson 50
Inserido (rusbe|Michael Joseph Jackson|50)

> ? rusbe
(rusbe|Michael Joseph Jackson|50)

> - afdg
Removido (afdg|John Winston Ono Lennon|40)

> ? afdg
ID afdg nao encontrado.

> + baby Justin Bieber 24
Inserido (baby|Justin Bieber|24)

> + bab Alok 32
Inserido (bab|Alok|32)

> ? baby
(baby|Justin Bieber|24)

> - dwiop
Removido (dwiop|George Harrison|58)

> P
(afd|James Paul McCartney|81)
(bab|Alok|32)
(baby|Justin Bieber|24)
(dafd|Larissa de Macedo Machado|30)
(dwiz|Richard Starkey|83)
(rusbe|Michael Joseph Jackson|50)

> S
banco.txt salvo

> F
```

Após essa execução, o arquivo `banco.txt` deve conter (os cadastros podem estar em qualquer ordem):

```
6
afd James Paul McCartney 81
bab Alok 32
baby Justin Bieber 24
dafd Larissa de Macedo Machado 30
dwiz Richard Starkey 83
rusbe Michael Joseph Jackson 50
```

Estrutura de Dados

Os cadastros devem ser armazenados e manipulados em uma **árvore de prefixos (*trie*)** de acordo com seus IDs. Cada nodo terminal deverá conter um ponteiro para uma *struct* contendo o cadastro completo. A figura abaixo exemplifica a *trie* contendo os 5 cadastros dados acima, como exemplo do arquivo `banco.txt` inicial:



- Todo comando (`?,+,-,P,S`) deve ser processado a partir da *trie*; assim, o comando `P`, por exemplo, deve percorrer a *trie* (e não alguma outra estrutura) para determinar todos os cadastros; o comando `?` deve fazer uma busca na *trie*; etc.
- Você pode assumir que todo ID tem no máximo 30 caracteres; que todo Nome Completo tem no máximo 1000 caracteres; e que toda idade é de até 99 anos;
- Não esqueça de liberar toda a memória utilizada por seu programa ao final da execução.

Orientações

- O trabalho pode ser feito por equipes de *no máximo* 2 (dois) estudantes;
- Submeta, via *Moodle*, um pacote **zip** ou **tar.gz** contendo todo o código-fonte necessário para compilar e executar sua solução, além de um arquivo de texto (txt) onde conste:
  - O nome de todos os integrantes da equipe;
  - Toda informação que a equipe julgar relevante para a correção (como *bugs* conhecidos, detalhes de implementação, escolhas de projeto, etc.)
- Comente adequadamente seus códigos para facilitar a correção;
- Atenção: a correção será parcialmente automatizada, e a saída do programa será testada com outras entradas além das fornecidas como exemplo. *Siga* ***fielmente*** *o formato de saída dado nos exemplos*, sob pena de grande redução da nota;
- Certifique-se que seu programa compila e funciona antes de submetê-lo;
- O trabalho deve ser entregue até **26 de Novembro de 2023, 23:59**, apenas via *Moodle*. Trabalhos entregues por outros meios ou fora do prazo não serão aceitos. É suficiente que o trabalho seja submetido por apenas um estudante da equipe;
- Trabalhos detectados como cópia, plágio (de colegas ou da internet) ou comprados receberão **todos** a nota 0 (**ZERO**) e estarão sujeitos a abertura de Processo Administrativo Disciplinar Discente.