Corretor Automático
SCC0201 - Trabalho 03

Moacir A. Ponti

September 12, 2018

1 Resumo

Implemente um programa que, dado um livro referência e uma lista de *tweets* da API do Twitter, imprima as palavras escritas de forma incorreta nos tweets (palavras que não existem no livro de referência).

2 Background

Depois de muitos momentos embaraçosos proporcionados pelo **Corretor Automático**, um grupo de inconformados conhecido como *Os Branquinhos* resolveu tomar uma providência. Depois de invadir as sedes das maiores empresas de tecnologia do mundo, *Os Branquinhos* destruíram todos os códigos e conhecimentos existentes sobre o **Corretor Automático**. No ápice de sua violência infundada, o grupo também incendiou diversos livros, incluindo os dicionários, para dificultar a a volta de um novo **Corretor Automático**.

Com a sociedade em colapso, a missão dos alunos de ICC2 é mostrar que a revolução é feita com cultura e ciência e não com violência, usando os conhecimentos da disciplina para dar início ao projeto de um novo **Corretor Automático**, leve e eficiente para rodar nos celulares que ainda restam no planeta.

Os chefes de estado determinaram que os alunos devem começar devagar, implementando **um programa que destaca as palavras incorretas em tweets, usando como referência palavras encontradas nos poucos livros (não-dicionários) restantes nas bibliotecas do mundo.**

3 Especificação

Implemente um programa chamado `spellcheck`, usando a linguagem `C`, que seja capaz de usar um grande conjunto de palavras (um livro) para verificar e imprimir as palavras incorretas (leia: **não encontradas no livro**) em uma lista de *tweets* passados como entrada

O nome do arquivo com o livro será passado na primeira linha da entrada e o nome do arquivo com o texto estará na segunda linha, da seguinte forma:

```
./spellcheck
dictionary.txt
tweets.json
```

3.1 Livro de Referência

O livro está encodado em ASCII e escrito em *plain text*, como no exemplo a seguir:

```
Alice was beginning to get very tired of sitting by her sister on the
bank, and of having nothing to do: once or twice she had peeped into the
book her sister was reading, but it had no pictures or conversations in
it, 'and what is the use of a book,' thought Alice 'without pictures or
conversations?'
```

Lembre-se de que seria muito ineficiente simplesmente guardar todas as palavras do livro e buscar sequencialmente. O aluno **deve** usar o livro para gerar um dicionário de "palavras conhecidas", sem palavras duplicadas, com uma estrutura de dados de sua escolha mas organizado de forma que uma busca de palvras seja eficiente.

3.2 Lista de Tweets em JSON

O arquivo com os tweets será exatamente a resposta da API para uma busca, em formato JSON, o formato está especificado com um exemplo na documentação do Twitter: `https://tinyurl.com/yavogrcj`. Note como a resposta é composta por um vetor de *tweets* sob a chave "statuses" e metadados sobre a busca sob a chave "search_metadata".

Cada *tweet* dentro desse vetor de *tweets* é descrito na documentação (`https://tinyurl.com/y6wdvjx6`) como seguindo o modelo fundamental:

```
{
     "created_at": "Thu May 10 15:24:15 +0000 2018",
     "id_str": "850006245121695744",
     "text": "Here is the Tweet message.",
     "user": {
     },
     "place": {
     },
     "entities": {
     },
     "extended_entities": {
     }
}
```

Onde a única chave importante para nós é o conteúdo da chave "text", a mensagem texto do *tweet*.

3.3 Saída do Programa

Seu código deve extrair o texto principal de todos os *tweets* e imprimir as palavras incorretas (não encontradas no livro de referência) na ordem do encontro, separadas uma da outra por espaço, separando a lista de palavras de tweets diferentes com um `\n`. Caso nenhuma palavra em um *tweet* seja considerada errada, imprima na tela apenas o `\n` para a linha equivalente a

este *tweet*. No exemplo a seguir, o usuário passou como livro referência o arquivo `alice_in_the_wonderland.txt` e a resposta da API contida no arquivo `tweets.json` descreve uma lista com três *tweets*. A resposta então tem três linhas com as palavras incorretas separadas por espaço:

```
> ./spellcheck
> alice_in_the_wonderland.txt
> tweets.json
Wat ar yah doin\n
Theyre\n
Dis iss mi jam\n
```

É possível que a chamada da API do Twitter tenha retornado um erro; neste caso, nenhum tweet estará presente e a saída do programa deve ser:

```
No tweets to check\n
```

Além destes requerimentos, fiquem de olho:

- O aluno **deve** usar o livro para gerar um "dicionário de palavras conhecidas", sem palavras duplicadas, evitando desperdício de memória.
- A **busca** no dicionário deve ser realizada de forma **eficiente** (não vale olhar palavra por palavra).
- Os textos dos *tweets* e dos livros podem conter pontuação que deve ser ignorada. Vamos testar se vocês estão ignorando `.`, `,`, `!`, `?`, `"`, `\n`, `&`, `...`, `'`, `'`, `;`, `@` e `#`.
- Os livros e os *tweets* estarão em inglês (para evitar acentos). A maior palavra existente em inglês é "Pneumonoultramicroscopicsilicovolcanoconiosis", com 45 caracteres.
- O maior número de caracteres permitidos em um *tweet* é 280 caracteres.
- Para facilitar, foram removidas URLs de dentro do corpo de *tweets*.
- O algorítimo de ordenação deve ser implementado pelo aluno e deve ter eficiência de pelo menos $\mathcal{O}(n \log n)$.
- Se alocação dinâmica for utilizada, seu programa não pode apresentar vazamento de memória.

3.4 Entrega e Avaliação

O trabalho será avaliado levando em consideração:

1. Realização dos objetivos / lógica utilizada
2. Uso de comentários e estrutura no código (e.g. indentação, legibilidade, modularização)
3. Resultado da plataforma run.codes
4. Eficiência da implementação

ATENÇÃO:

- O projeto deverá ser entregue apenas pelo (`http://run.codes`) no formato de **código fonte**, ou seja apenas o código C.
- O prazo está no sistema run.codes
- Em caso de projetos **copiados** de colegas ou da Internet, todos os envolvidos recebem nota zero. Inclui no plágio a cópia com pequenas modificações, cópia de apenas uma parte ou função. Portanto programe seu próprio trabalho.