INSTITUTO SUPERIOR TÉCNICO

Introdução aos Algoritmos e Estruturas de Dados
2012/2013

Enunciado do 2º Projecto
Data de entrega: Segunda-feira, 13 de Maio de 2013 (23h59)

1 Introdução

O objectivo deste projeto é o desenvolvimento, em linguagem C, de um programa para análise das palavras e linhas de um texto. Uma vez que as funcionalidades pretendidas são conceptualmente muito simples, pretende-se que os alunos dediquem particular esforço à implementação de soluções computacionalmente eficientes, o que será valorizado em termos da avaliação. Assim, não será apenas tida em consideração a correção do código produzido, mas também a eficiência do mesmo.

2 Especificação do Programa

O programa a implementar deverá ler um texto e permitir realizar um conjunto de operações de análise do mesmo. Um texto consiste num conjunto de linhas, separadas por um carácter de mudança de linha. Uma linha consiste num conjunto de palavras separadas por um ou mais caracteres separadores. Uma linha pode ter no máximo 1025 caracteres, contando com o carácter de mudança de linha. Os caracteres separadores são:

- espaço;
- tab;
- virgula;
- ponto-e-virgula;
- ponto final;
- ponto de interrogação;
- ponto de exclamação;
- aspas;
- mudança de linha.

Uma palavra é uma sequência constituída por quaisquer caracteres, exceptuando os caracteres separadores. Uma palavra pode conter no máximo 1024 caracteres. Dado um texto, deverão ser implementadas as seguintes funcionalidades:

- esquecer uma palavra;
- mostrar o texto;
- mostrar todas as linhas em que ocorre uma determinada palavra;
- mostrar todas as linhas em que ocorrem duas palavras, e estão em posições adjacentes;
- mostrar uma listagem do número de ocorrências de cada palavra.

Após um palavra ter sido esquecida esta não deve ser contabilizada nos *outputs* dos comandos subsequentes.

3 Dados de Entrada e Saída

O programa deverá ler os dados de entrada a partir do *standard input*. Os dados de entrada são constituídos por duas secções: texto e comandos. A secção de texto é iniciada com uma linha contendo apenas o número de linhas do texto, sendo as linhas seguintes o texto propriamente dito. A secção de comandos é constituída por um número arbitrário de linhas, onde cada linha é iniciada por um carácter que representa cada comando, seguido de um número variável de parâmetros; o comando e cada um dos parâmetros são separados por um espaço.

Passamos agora a detalhar os dados de entrada e saída de cada um dos comandos disponíveis — `f`, `s`, `l`, `w` e `h`. Cada comando indica uma determinada ação que se passa a descrever, em termos de objectivo, sintaxe e *output*. Em todos os comandos que recebam como argumento uma palavra, as letras maiúsculas devem ser consideradas equivalentes às respectivas letras minúsculas (*case insensitive*).

3.1 Esquecer uma palavra

A palavra indicada não aparecerá mais no *output* resultante de qualquer dos restantes comandos, exceptuando do comando `s`.

Sintaxe:
`f <palavra>`

- `<palavra>`: palavra a esquecer, contendo, no máximo, 1024 caracteres.

Output: nenhum.

3.2 Mostrar texto

Mostra o texto. Deverá ser mostrado o texto original, contendo inclusivamente as palavras que foram entretanto esquecidas e que não devem ser mostradas nos restantes comandos.

Sintaxe:
`s`

- não tem argumentos.

Output: todas as linhas que constituem o texto original.

3.3 Mostrar ocorrências de uma palavra

Mostra todas as linhas onde ocorre uma determinada palavra, precedidas pelo respectivo número de linha e um espaço. As linhas devem ser mostradas por ordem crescente do respectivo número de linha. Se a palavra tiver sido esquecida com o comando `f`, nada deve ser mostrado.

Sintaxe:
`l <palavra>`

- `<palavra>`: palavra a procurar, contendo, no máximo, 1024 caracteres.

Output:
`<número> <linha>`
…
`<número> <linha>`

- `<número>`: número da linha, que é um número inteiro positivo a começar em 1;
- `<linha>`: conteúdo da linha.

3.4 Mostrar ocorrências de duas palavras consecutivas

Mostra todas as linhas onde ocorrem duas palavras em posições consecutivas, precedidas pelo respectivo número de linha e um espaço. A ordem das palavras é irrelevante. Por posições consecutivas entende-se uma palavra estar imediatamente antes ou depois da outra, e na mesma linha, no texto original, sendo apenas separadas por caracteres separadores. Se uma ou ambas as palavras tiverem sido esquecidas com o comando `f`, nada deve ser mostrado. As linhas devem ser mostradas por ordem crescente do respectivo número de linha.

Sintaxe:
`w <palavra> <palavra>`

- `<palavra>`: palavras a procurar, contendo as duas, no máximo, 1023 caracteres.

Output:
`<número> <linha>`
…
`<número> <linha>`

- `<número>`: número da linha, que é um número inteiro positivo a começar em 1;
- `<linha>`: conteúdo da linha na sua versão original.

3.5 Mostrar listagem de número de ocorrências de cada palavra

Mostra uma listagem, em que cada linha contém uma palavra, seguida por um espaço e pelo número de ocorrências da mesma no texto. As palavras devem ser mostradas por ordem lexicográfica (alfabética). As palavras que tenham sido esquecidas com o comando `f`, não devem ser mostradas na listagem.

Sintaxe:
`h`

Output:
`<palavra> <número>`
…
`<palavra> <número>`

- `<palavra>`: palavra em letras minúsculas, contendo, no máximo, 1024 caracteres;
- `<número>`: número de ocorrências da palavra no texto, que é um número inteiro positivo a começar em 1.

4 Exemplos

4.1 Exemplo 1

Input:

```
4
As armas e os baroes assinalados,
Que da ocidental praia Lusitana,
Por mares nunca de antes navegados,
Passaram ainda alem da Taprobana,
s
l da
h
```

Output:

```
As armas e os baroes assinalados,
Que da ocidental praia Lusitana,
Por mares nunca de antes navegados,
Passaram ainda alem da Taprobana,
```

output do comando `s`

```
2 Que da ocidental praia Lusitana,
4 Passaram ainda alem da Taprobana,
```

output do comando `l da`

```
ainda 1
alem 1
antes 1
armas 1
as 1
assinalados 1
baroes 1
da 2
de 1
e 1
lusitana 1
mares 1
navegados 1
nunca 1
ocidental 1
os 1
passaram 1
por 1
praia 1
que 1
taprobana 1
```

output do comando `h`

4.2 Exemplo 2

Input:

```
9
Alma minha gentil, que te partiste
Tao cedo desta vida descontente,
Repousa la no Ceu eternamente,
E viva eu ca na terra sempre triste.

Se la no assento eterio, onde subiste,
Memoria desta vida se consente,
Nao te esquecas daquele amor ardente
Que ja nos olhos meus tao puro viste.
f la
w na ca
w la no
l que
s
```

Output:

```
4 E viva eu ca na terra sempre triste.
1 Alma minha gentil, que te partiste
9 Que ja nos olhos meus tao puro viste.
Alma minha gentil, que te partiste
Tao cedo desta vida descontente,
Repousa la no Ceu eternamente,
E viva eu ca na terra sempre triste.

Se la no assento eterio, onde subiste,
Memoria desta vida se consente,
Nao te esquecas daquele amor ardente
Que ja nos olhos meus tao puro viste.
```

output do comando `w na ca`

output do comando `l que`

output do comando `s`

5 Parsing

Apresenta-se aqui uma sugestão de código a utilizar para efectuar a divisão do texto em palavras.

```
#define NUMSEP 9
const char separators[] = { ' ','\t',',',';','.','?','!','"','\n' };

void split(char *line)
{
  int i, j, k;
  char buffer[MAXLINESIZE];
  for(i = 0, k = 0; line[i] != '\0'; i++, k++) {
    buffer[k] = tolower(line[i]);
    for(j = 0; j < NUMSEP; j++) {
      if(line[i] == separators[j]) {
        if(k != 0) {
          buffer[k] = '\0';
          /* processar aqui palavra guardada em buffer */
        }
        k = -1;
      }
    }
  }
}
```

Nota: Em alternativa, poderá utilizar a função `strtok` disponível em *string.h*, assim como as restantes funções deste *header* da *C standard library*.

6 Compilação do Programa

O compilador a utilizar é o `gcc` com as seguintes opções de compilação: `-ansi -Wall -pedantic`. Para compilar o programa deve executar o seguinte comando:

```
$ gcc -ansi -Wall -pedantic -o proj2 *.c
```

o qual deve ter como resultado a geração do ficheiro executável `proj2`, caso não haja erros de compilação. A execução deste comando não deverá escrever qualquer resultado no terminal. Caso a execução deste comando escreva algum resultado no terminal, considera-se que o programa não compilou com sucesso. Por exemplo, durante a compilação do programa, o compilador não deve escrever mensagens de aviso (*warnings*).

7 Execução do Programa

O programa deve ser executado da forma seguinte:

```
$ ./proj2 < test1.in > test1.myout
```

Posteriormente poderá comparar o seu output com o output previsto usando o comando `diff`

```
$ diff test1.out test1.myout
```

8 Entrega do Projecto

A entrega do projecto deverá respeitar o procedimento seguinte:

- Na página da disciplina aceda ao sistema para entrega de projectos. O sistema será activado uma semana antes da data limite de entrega. Instruções acerca da forma de acesso ao sistema serão oportunamente fornecidas.
- Efectue o upload de um ficheiro arquivo com extensão `.zip` que inclua os ficheiros fonte (`.c`) e cabeçalho (`.h`) que constituem o programa.
- Para criar um ficheiro arquivo com a extensão .zip deve executar o seguinte comando na directoria onde se encontram os ficheiros com extensão .c e .h, criados durante o desenvolvimento do projecto:
  ```
  $ zip proj2.zip *.c *.h
  ```
- Como resultado do processo de upload será informado se a resolução entregue apresenta a resposta esperada num conjunto de casos de teste.
- O sistema não permite submissões com menos de 10 minutos de intervalo para o mesmo grupo. Exemplos de casos de teste serão oportunamente fornecidos.
- Data limite de entrega do projecto: 13 de Maio de 2013 (23h59m). Até à data limite poderá efectuar o número de entregas que desejar, sendo utilizada para efeitos de avaliação a última entrega efectuada. Deverá portanto verificar cuidadosamente que a última entrega realizada corresponde à versão do projecto que pretende que seja avaliada. Não serão abertas excepções.

9 Avaliação do Projecto

9.1 Componentes da Avaliação

Na avaliação do projecto serão consideradas as seguintes componentes:

1. A primeira componente avalia o desempenho e funcionalidade do programa realizado. Esta componente é avaliada entre 0 e 16 valores.

2. A segunda componente avalia a qualidade do código entregue, nomeadamente os seguintes aspectos: comentários, indentação, estruturação, modularidade, abstracção, entre outros. Esta componente poderá variar entre -4 valores e +4 valores relativamente à classificação calculada no item anterior e será atribuída na discussão final do projecto. Nesta componente será também utilizado o sistema *valgrind* for forma a detectar fugas de memória ("memory leaks") ou outras incorrecções no código, que serão penalizadas. Aconselha-se por isso que os alunos utilizem este sistema para fazer debugging do código e corrigir eventuais incorrecções antes da submissão do projecto.

3. Durante a discussão final do projecto será averiguada a participação de cada elemento do grupo na realização do projecto, bem como a sua compreensão do trabalho realizado, sendo a respectiva classificação ponderada em conformidade, isto é, elementos do mesmo grupo podem ter classificações diferentes. Elementos do grupo que se verifique não terem participado na realização do respectivo projecto terão a classificação de 0 (zero) valores.

9.2 Atribuição Automática da Classificação

- A classificação da primeira componente da avaliação do projecto é obtida automaticamente através da execução de um conjunto de testes executados num computador com o sistema operativo GNU/Linux. Torna-se portanto essencial que o código compile correctamente e que respeite o formato de entrada e saída dos dados descrito anteriormente. Projectos que não obedeçam ao formato indicado no enunciado serão penalizados na avaliação automática, podendo, no limite, ter 0 (zero) valores se falharem todos os testes. Os testes considerados para efeitos de avaliação podem incluir (ou não) os disponibilizados na página da disciplina, além de um conjunto de testes adicionais. A execução de cada programa em cada teste é limitada na quantidade de memória que pode utilizar, até um máximo de 64 Mb, e no tempo total disponível para execução, sendo o tempo limite distinto para cada teste.

- Note-se que o facto de um projecto passar com sucesso o conjunto de testes disponibilizado na página da disciplina não implica que esse projecto esteja totalmente correcto. Apenas indica que passou alguns testes com sucesso, mas este conjunto de testes não é exaustivo. É da responsabilidade dos alunos garantir que o código produzido está correcto.

- Em caso algum será disponibilizada qualquer tipo de informação sobre os casos de teste utilizados pelo sistema de avaliação automática. A totalidade de ficheiros de teste usados na avaliação do projecto serão disponibilizados na página da disciplina após a data de entrega.