

# Trabalho de shell

## 09/02/2023 V 1.1

### 1 Requisitos obrigatórios

Seus *scripts* devem:

1. se chamar `tshell_p1.sh`, `tshell_p2.sh` e `tshell_p3.sh`;
2. estar com a hashbang e ter permissão 700;
3. os professores farão a correção em uma destas máquinas: `cpu1`, `cpu2` ou `orval`; portanto você deve garantir que seu script possa ser rodado nelas;
4. fazer uso de boas práticas de programação, como indentação, variáveis, bons nomes para variáveis e funções, comentários no código, ...
5. fazer uso de pelo menos uma boa função, isto é uma função que se justifique, ou evitando repetição de código ou para fins de legibilidade, ou ambos;
6. seu script não deve “poluir” a tela com mensagens dos comandos, a saída deve ser exatamente o que pede o enunciado abaixo;
7. ao término da execução do seu script, o usuário que o executou deve estar com seu prompt no mesmo diretório em que estava ao executá-lo.
8. a entrega do trabalho deve ser um arquivo `.tar.gz` contendo um diretório chamado `tshell` com os 3 *scripts*.

Leia cuidadosamente as seções do restante deste documento. Em caso de dúvida entre em contato. Não deixe de verificar possíveis atualizações desse documento.

### 2 O problema

O estudo de moléculas envolvidas nas vias metabólicas de células e tecidos humanos é um tema de interesse de várias áreas de pesquisa nas ciências da saúde, em particular o estudo de tumores e processos neoplásicos.

Considerando que muitos grupos de pesquisa, no mundo todo, estudam e publicam o papel de várias moléculas nesses processos, listar e contabilizar os

estudos publicados que relacionam essas moléculas com determinados tipos de tumores é uma etapa preliminar importante da pesquisa na área. Os nomes de tumores ou de tipos de células são geralmente usados como termos de filtragem para tornar o processo de busca mais eficiente, reduzindo o tamanho do conjunto de dados a ser analisado. Sobre o conjunto reduzido, então se busca as moléculas de interesse. Em especial, as moléculas ditas “fatores de transcrição” são de grande importância, pois atuam diretamente na expressão de genes ligados aos processos neoplásicos.

O processo de busca e totalização de artigos científicos que contenham termos de interesse em linguagem natural pode ser descrito pelo procedimento que segue:

- a. Aquisição do conjunto de dados: nessa etapa são copiados os dados da base de referências científicas PubMed<sup>1</sup>, que é a principal base aberta de dados estruturados de citações científicas na área biomédica. A base de dados disponível para cópia contém cerca de 34 milhões de entradas em formato XML<sup>2</sup> e ocupa aproximadamente 350 Gigabytes de espaço de armazenamento;
- b. Filtragem do conjunto de dados: nessa etapa são extraídos do conjunto obtido na etapa anterior apenas os campos relevantes para o processo de busca: título, resumo e palavras chaves. O processo de limpeza visa reduzir o espaço de armazenamento necessário para os dados;
- c. Definição dos termos de interesse: nessa etapa são definidos os termos em linguagem natural utilizados no processo de busca. Por exemplo, os termos de filtragem primários podem ser: “stem cell”, “stem like cell” e “glioblastoma”. Além dos termos primários, é definido um conjunto secundário com 1624 identificadores de fatores de transcrição de interesse;
- d. Aquisição do conjunto de sinônimos: nessa etapa são definidos conjuntos de identificadores equivalentes para cada fator de transcrição de interesse. Para a construção dos conjuntos de sinônimos para cada fator de transcrição é utilizada a base Gene\_Info disponibilizada pela NCBI - *National Center for Biotechnology*<sup>3</sup>. Com a construção das tabelas de sinônimos para cada fator de transcrição, o conjunto de identificadores a serem usados como termos secundários no processo de busca passa de 1624 para 24967 identificadores;

---

<sup>1</sup><https://pubmed.ncbi.nlm.nih.gov>

<sup>2</sup><https://dtd.nlm.nih.gov/ncbi/pubmed/doc/out/190101/index.html>

<sup>3</sup><https://www.ncbi.nlm.nih.gov/gene>

- e. Busca pelos termos primários: essa etapa extrai todas as entradas do conjunto de dados que contenham todos os termos primários definidos na etapa c. O processo de filtragem consiste em verificar a ocorrência de cada termo primário em algum dos campos da base obtida na etapa b. São ignoradas diferenças entre letras maiúsculas e minúsculas, pontuação e separadores;
- f. Busca pelos fatores de transcrição: para cada fator de transcrição definido na etapa c, são extraídos nesta etapa os artigos que contêm algum dos identificadores presentes na tabela de sinônimos do fator de transcrição. O processo de busca é aplicado sobre os dados resultantes da etapa e. São obtidos, para cada fator de transcrição, o conjunto de artigos que contêm os termos primários e algum identificador sinônimo do fator de transcrição;
- g. Totalização dos resultados: para cada fator de transcrição, é gerado como saída o número de artigos encontrados que contêm todos os termos primários e algum sinônimo do fator de transcrição.

## 3 A tarefa

O trabalho está dividido em três partes.

### 3.1 Parte 1

Escreva um *script* em bash que execute parte do procedimento descrito no problema, da seguinte forma:

1. Uma parte da base PubMed foi copiada para o sistema de arquivos do DInf no diretório

`/nobackup/prof/fabiano/PubMed`

Cada arquivo que termina com `.xml.gz` é um texto, compactado com o programa `gzip`, que contém um subconjunto de publicações da base no formato XML. Este formato tem uma estrutura hierárquica bastante complexa, com os vários atributos de cada publicação. Os arquivos que terminam com `.csv` são versões simplificadas do mesmo conteúdo, contendo apenas os atributos relevantes para a busca: identificador do artigo, título, resumo e palavras chave. Note que nos arquivos `.csv`: cada publicação ocupa uma linha; a primeira linha indica os campos

que constituem cada linha e não deve ser processada; o separador entre os campos é o caracter < e; apenas o último campo de cada linhas pode estar vazio.

Uma observação importante é que no arquivo `.csv` todas as publicações têm pelo menos os campos título e resumo contendo textos.

Assim, as duas primeiras etapas do procedimento do problema já estão concluídas e o seu *script* pode processar diretamente os arquivos `.csv`, sem se preocupar com os arquivos XML complicados.

2. O segundo conjunto de dados de entrada para o seu *script* é composto pelos termos primários de busca. Cada termo pode ser composto por uma sequência de palavras separadas por espaços. Os termos são passados como parâmetros na linha de comando, na chamada do seu *script*. Assuma que o número máximo de termos primários é 3.
3. A chamada do seu *script* deve seguir o seguinte formato:

```
./tshell_p1.sh <dpm> <ds> <t1> [<t2> [<t3>]]
```

sendo os itens entre [ ] opcionais. <dpm> é o diretório que contém os arquivos de entrada `.csv` com os dados da base PubMed, <t1>, <t2> e <t3> são os termos primários de busca e <ds> é o diretório onde você deve escrever os arquivos de saída.

4. Para cada arquivo `.csv` do diretório de entrada, o seu *script* deve extrair as linhas que contém todos os termos primários de busca e escrever as linhas extraídas em arquivos como o mesmo nome no diretório de saída. Todas as linhas dos arquivos resultantes no diretório de saída conterão pelo menos uma ocorrência de cada termo primário de busca.

## 3.2 Parte 2

Escreva um *script* em bash que execute a segunda parte do procedimento descrito no problema da seguinte forma:

1. O primeiro conjunto de dados para o seu *script* é dado por um diretório contendo arquivos `.csv` no formato descrito no item 1 da parte 1 do trabalho.
2. O segundo conjunto de dados de entrada para o seu *script* está no diretório

/nobackup/prof/fabiano/Sinonimos

O nome de cada arquivo nesse diretório é o identificador de um fator de transcrição seguido de `.txt`. O conteúdo de cada arquivo é uma lista (em ordem lexicográfica, um por linha, em minúsculas) de identificadores que são sinônimos do fator de transcrição. Note que o identificador que dá nome ao arquivo também está na lista de sinônimos dentro do arquivo.

Esses arquivos resolvem a etapa d do procedimento, basta usar esses arquivos como tabelas de sinônimos para cada fator de transcrição.

3. O seu *script* deve gerar como saída na tela uma lista, um por linha, dos fatores de transcrição e o número correspondente de artigos encontrados que contêm algum dos sinônimos do fator.

Na saída, o nome do fator de transcrição deve ser o mesmo que nomeia o arquivo de sinônimos correspondente, sem o `.txt`. Cada nome deve vir seguido de `:` e imediatamente do número de artigos encontrados. A lista deve estar ordenada na ordem lexicográfica. Por exemplo:

```
A4GALT:0
AATF:7
ABCA1:35
ABCA8:1
ABCB10:0
...
```

4. A chamada do seu *script* deve seguir o seguinte formato:

```
./tshell_p2.sh <dcsv> <dsin>
```

sendo `<dcsv>` o diretório que contém os arquivos `.csv` com os dados da base PubMed e `<dsin>` o diretório que contém os arquivos com as tabelas de sinônimos para cada fator de transcrição.

### 3.3 Parte 3

Escreva um *script* em bash que converta o arquivo `.xml.gz` no seu arquivo `.csv` correspondente.

Para extrair os campos indicados na etapa de limpeza, use o programa `xgrep` que está instalado em

```
/home/soft/xgrep/bin
```

Por exemplo:

```
xgrep -t -x "//PMID|//ArticleTitle|//Abstract|//KeywordList" a.xml
```

extraí os 4 campos necessários de um arquivo `a.xml` que contém um conjunto de publicações do PubMed. Para entender melhor como funciona o `xgrep` leia o manual:

```
man -l /home/soft/xgrep/share/man/man1/xgrep.1
```

Note que a saída gerada pelo `xgrep` não está no formato CSV esperado. Sua tarefa é contruir um *script* que transforme a saída do `xgrep` no formato CSV descrito no item 1 da parte 1 do trabalho.

A chamada do seu *script* deve seguir o seguinte formato:

```
./tshell_p3.sh <axml> <acsv>
```

sendo `<axml>` o arquivo de entrada no formato XML da base PubMed e `<acsv>` o arquivo de saída no formato CSV descrito no item 1 da parte 1 do trabalho.

Use os arquivos fornecidos no diretório

```
/nobackup/prof/fabiano/PubMed
```

para testar seu *script*. A saída deve ser idêntica aos arquivos `.csv` fornecidos.

Note que em cada linha do arquivo de saída os campos `version+pmid`, `title` e `abstract` não podem estar vazios.

## 4 Dicas e comandos úteis

Revise seus *scripts* cuidadosamente e, se necessário, refaça partes deles para que sejam mais compreensíveis e elegantes. Finalmente, faça sua entrega dentro do prazo estipulado.

Aqui tem uma lista não exaustiva de comandos que talvez sejam úteis. Não se limite a eles, pode ser que você encontre soluções alternativas e, quem sabe, melhores.

```
grep
awk
cat
cut
for
```

```
head
popd
pushd
sed
tr
for
if
[ ]
```

## 5 Conceitos úteis

Você provavelmente vai precisar usar os conceitos apresentados, tais como (lista não exaustiva):

- Substituição de comandos
- Variáveis
- Símbolos especiais
- Entrada, saída e pipes
- Os comandos: `for`, `if`, `[ ]`, funções

Um comando interessante para ajudar a depurar *scripts shell* é você colocar no início do *script* o seguinte:

```
set -x
```

Depois, comente esta linha para a entrega do trabalho.

## 6 Conceitos novos

Faz parte do trabalho você descobrir algumas coisas novas. Para isso, consulte as *man pages* dos comandos indicados ou de outros que você encontre. Vários sites na internet são muito bons para encontrar soluções. Um bom exemplo é o GeekForGeeks ou o StackOverflow. Aprender a buscar informações faz parte da atividade de programação de computadores em geral. Você deve estar preparado para isso.