**Analisadores Léxicos e Sintáticos**
**2021-1 – Roteiro de Aula de Laboratório : SED**

Prof. Edward Hermann Haeusler
Bernardo Alkmim 10 de março de 2021

Este roteiro está dividido em duas partes. Na primeira parte alguns exemplos de uso de expressões regulares com captura serão apresentados e você deve ser capaz de prever o resultado da execução. Procure antever o resultado, se você simplesmenre ignorar e partir para a execução do exemplo estará se sabotando na aula. Dê preferência a interação com seus colegas ou com o professor para dirimir dúvidas e melhorar seu desempenho nesta epata. Na segunda parte do roteiro aguns problemas serão propostos e você terá que propor soluções com o uso de SED e executar estas solucões nos respectivos arquivos.

# 1 Primeira parte

O comando mais simples e popular do **sed** é o **s**: *sed s/palavra*1*/palavra*2/, que substitui o padrão especificado pela *palavra*1 pela *palavra*2 na entrada. O caso mais simples é com *palavra*1 sendo uma cadeia simples de caracteres. Indique qual o resultado da execução dos comandos abaixo. Lembre dos redirecionamentos feitos com os símbolos $<$ e $>$ e, assim como o pipe |. Use algum arquivo texto no lugar de textfile.

```
sed s/day/night/ Sunday
echo Sunday | sed s/day/night/
sed s/day/night/ < textfile
```

Lembre-se que **sed** é orientado por linha, i.e, ele executa cada comando uma vez em cada linha. Examine o arquivo **exercicio1**. Qual você espera que seja o resultado do comando '*sed s/one/ONE/* $<$ *exercicio*1' Após responder a esta pergunta, verifique sua resposta executando o comando. Você acertou ? Caso não tenha acertado verifique com o professor seu entendimento sobre o **sed**. O caracter / é um delimitador que tem significado especial para o **sed**, se você quiser um padrão que inclua este caracter, como por exemplo uma PATH de um diretório/arquivo, use o caracter \. Ele é um caracter de 'escape' que tem função 'quotar' o caracter seguinte. Isto é, \/ é interpretado pelo **sed** como simplesmente /. Oque você espera que o comando abaixo realize em um arquivo que seja dado a ele como entrada.

```
sed s/\/usr\/local\/bin/\/common\/bin/  arquivo
```

O caracter & é usado para capturar o padrão que foi reconhecido. Por exemplo, em um comando como abaixo a expressão regular [0-9]+ casa com qualquer sequência de digitos com pelo menos um digito. O **sed** captura a primeira ocorrência que casa com esta expressão e associa a &. O replacement do comando é "& &", o que você espera da execução deste comando sobre uma sequencia de caracteres que contenha números ?

```
sed -r s/[0-9]+/\& \&/
```

Obs: O -r no comando acima indica que estamos usando expressões regulares estendidas. Isto nos permite usar E+ no lugar de EE*. Usuários de FreeBSD e/ou MACOSX devem usar -E (ou -e) no lugar de -r.

Importante: Quando se escreve o comando sed entre 's não há necessidade de usar a \ para fazer o 'escape' Experimente os dois tipos de comandos.

Para capturar parte de uma expressão regular se usa os delimitadores \( e \). Para fazer referência ao padrão capturado se usa a expressão $\backslash n$, onde n é a ordem da expressão capturada. Por exemplo, em

```
sed -r s/\([0-9]+\)K\([a-z]\)/\\2\\1/
```

o **sed** procura sequências de digitos seguidos da letra "K" seguidos alguma outra letra (o padrão [a-z]), substituindo o padrão pelos respectivos sub-padrões na ordem invertida. O que acontece se o comando acima for executado sobre o string '9123Ka' ?? Observe que o comando acima tem o mesmo efeito de

```
sed -r 's/([0-9]+)K([a-z])/\2\1/'
```

Utilize o arquivo exercicio2 para executar os exemplos acima.

Observe que os strings capturados não precisam estar no string the 'replacement', isto indica que eles não aparecerão no resultado do **sed**. Por exemplo, o comando

```
sed -E 's/([a-zA-Z]+)K([0-9]+)/\2/'
```

remove palavras antes de K quando após o K há um número (numeral). Por outro lado, pode-se usar o padrão capturado em outra captura subsequente. O comando:

```
sed -E 's/([a-zA-Z]+]\1/\1/'
```

Remove um duplo de cada palavra duplicada de um arquivo.

Pode-se usar o **sed** como um filtro. Neste caso o comando é

```
sed -E '/([a-zA-Z]+)\1/p'
```

que mostrará todas as linhas que contenham palavras duplicadas seguidas. Verifique usando 'man sed' todas as opções de filtro e substituições.

# 2 Segunda Parte

Crie comando **sed** para cada uma das tarefas abaixo:

1. Em um arquivo csv que descreve uma série de medidas de um evento climático de diferentes partes do mundo existem temperaturas nas três escalas conhecidas (Kelvin, Celsius e Farenheit). Cada linha possui valores na mesma escala. Você reconhece a escala pelo símbolos $K$, $C$ ou $F$ após

o valores numérico decimal. A quantidade de espaços entre o valores e a letra que indica a escala é arbitrária e pode variar de linha para linha. Escreva um comando para cada escala, isto é, seu comando vai capturar e imprimir na saída padrão somente as temperaturas na escala escolhida, com a escala.

2. Uma série temporal de um dia da bolsa de valores é um arquivo csv, que tem o nome (sigla) da ação e o valor da mesma naquele instante. Faça um filtro que gere um outro arquivo com somente os valores de uma ação específica que é editada no comando. OBS:Pode ser necessário usar a opção global do sed (opção "g"). Consulte a documentação do sed (use o man).

3. Para um arquivo texto, detecte e substitua por XXX todas as linhas que possuam palavras que ocorrem pelo menos 3 vezes na linha.

4. Em uma lista de frases simples da forma Nome1 Verbo Artigo Nome2 onde Nome1 e Nome2 são substantivos, Verbo é um padrao na forma Radical+ou (por exemplo Pintou, Mostrou, Olhou, etc). Artigo é um artigo. A idéia é passar as frases para a voz passiva. Por exemplo 'Eduardo Pintou a escada' se transforma em 'A escada foi pintada por Eduardo'. Seu comando sed não precisa resolver todos os casos perfeitamente, mas a maioria dos casos com verbos transitivos diretos como os acima deve ser resolvidos.

5. No ítem 2 acima, imprima todos os valores acima de um certo valor para um certo ativo (ambos editados no comando).

6. Como você faz para generalizar o ítem acima de forma que o ativo e o valor limite sejam lidos e um comando sed seja criado para servir de filtro.

7. Obs: O ponto '.' é um padrão que casa com qualquer caracter que não for o $\backslash n$. Escreva uma expressão regular que detecte uma linha de texto que termina com $\backslash n$. Use-a em um comando sed para realizar a tarefa de escrever a palavra 'FIM' no fim de cada linha.

8. O ^ é um caracter que casa com o início de linha em arquivos texto, enquanto $ casa com final de linha. Escreva a funcionalidade do filtro acima com o uso do $. Escreva um comando sed que recebendo o resultado de um comando 'ls -l ' imprime a lista com onoem dos diretórios que estão neste 'ls -l '.

9. Escreva um filtro que detecte endereços válidos (bem formados sintaticamente) de email.

10. Escreva um comando **sed** que troque os endereços de email que ocorrem em um texto pela string [email addr].

11. Como na tarefa acima, mas no lugar de [email addr] escreva a string [email de nome], onde nome é a palavra que ocorre antes do @ no email.

12. Escreva um comando sed que atua sobre o código de um programa C e inicializa com o valor -1 todas as variáveis do tipo inteiro declaradas no programa. Inicialmente ignore que as declarações podem ter quebra de linha, depois faça incluindo esta complexidade adicional. Um exemplo simples é como abaixo, em que a segunda linha é o resultado do comando sobre a primeira linha.

```
Int X, YY, ZZw;
```

gera

```
Int X=-1, YY=-1, ZZw=-1;
```

13. Tente fazer um filtro para reconhecer expressões da linguagem $\{a^n b^n : n \geq 0\}$. Por que você não consegue cumprir esta tarefa ?

# Boa Prática