Projeto

1. Objetivo

O objetivo deste projeto é que os alunos apliquem conceitos aprendidos no decorrer do curso na criação de um *shell script* que resolva algum problema do mundo real que seja interessante e útil.

2. Conceitos

A tabela a seguir detalha 20 (vinte) conceitos que poderão ser utilizados no *shell script* que será desenvolvido. Embora o *script* possa fazer uso de outros conceitos alheios aos listados abaixo, serão considerados na contabilização da nota somente aqueles presentes nessa tabela.

#	Conceito	Aula
01	Comandos: `echo`, `ls`, `pwd`, `cd`, `tree`, `find`, `locate`, `whereis`, `file`, `stat`, `date`, `cal`, `history`, `clear`, `logout` ou `exit`	Comandos básicos
02	Comandos: `strings`, `cat`, `tac`, `wc`, `more`, `less`, `head`, `tail`, `tee`, `sort`, `zcat` ou `diff`	Processamento de texto
03	Comandos: `mkdir`, `cd`, `rmdir`, `mv`, `cp`, `mv`, `ln`, `id`, `umask`, `basename`, `dirname`, `gzip`, `gunzip`, `tar`, `split`, `shred`, `cksum`, `md5sum`, `dd`, `lsof` ou `fuser`	Sistema de arquivos
04	Permissões em representação octal ou simbólica: `chmod`	
05	Comandos: `ps`, `top`, `jobs`, `bg`, `fg`, `pstree`, `vmstat`, `xload` ou `tload`	Processos
06	Sinais: `kill` ou `killall`	
07	Redirecionamentos da entrada, saída ou saída de erro padrão	Entradas, saídas e Redirecionamentos
08	*Pipes* ou redirecionamentos avançados	
09	Expansão de arquivos, til (~), aritmética, de chaves, de parâmetros, substituição de comandos e escapes	Expansões
10	Variáveis: criadas pelo usuário ou reservadas	Parâmetros e Variáveis
11	Parâmetros posicionais e especiais: `$0` até `$n`, `$*`, `$@`, `$#`, `$?`, `$$` ou `$!`	
12	Atributos de variáveis: arranjo, função, inteira, somente escrita (`readonly`) ou global (`export`)	
13	Expansões avançadas: `${name:-value}`, `${name:=value}`, `${name:?value}`, `${name:+value}`, `${parameter:offset}`, `${parameter:offset:length}`, `${#parameter}`, `${!parameter}`, `${parameter#word}`, `${parameter##word}`, `${parameter%word}`, `${parameter%%word}`, `${parameter/pattern/string}` ou `${parameter//pattern/string}`	
14	Testes condicionais com `test` ou `if`: expressões com arquivos, strings, inteiros ou expressões regulares (=~)	Estruturas de fluxo condicionais
15	Testes condicionais com `case`	
16	Outras formas de desvio: `&&` ou `||`	
17	Comandos ou sequenciadores: `for`, `while`, `until`, `select`, `break` ou `continue`	Estruturas de fluxo de repetição

18	Expressão regular básica ou estendidas: `grep` ou `egrep` OBS: a expressão regular deve usar pelo menos um metacaractere	Expressões Regulares
19	Programa em `sed`	`sed`
20	Programa em `AWK`	`AWK`

3. Instruções

O projeto será feito em três partes de acordo com o calendário a seguir.

Etapa	Descrição	Datas
01	Definição do problema a ser resolvido no projeto.	14/10/2025
02	Relatório contendo a descrição do problema resolvido, código do *shell script* desenvolvido e planilha com os conceitos utilizados no formato definido.	01/12/2025
03	Apresentação do projeto para a turma.	04/12/2025 a 18/12/2025

Na primeira etapa, cada aluno deverá enviar para a lista de discussão (decom042@googlegroups.com) uma descrição do problema, interessante e útil, que se deseja resolver. **O problema escolhido deve, obrigatoriamente, obter informações de alguma fonte externa (arquivo, internet, etc) e/ou produzir alguma informação externa (documento, relatório, etc). Você não pode ler entradas de usuário do teclado com o comando `read` ou similares.**

Na segunda etapa, cada aluno deverá **preparar um relatório em formato PDF** que será submetido pelo SIGAA (https://sig.cefetmg.br/sigaa). O relatório deverá conter: (1) a descrição do problema que foi definido na lista de discussão, (2) o código do *shell script* desenvolvido para resolver esse problema com suas linhas numeradas, e (3) uma planilha com 15 (quinze) conceitos dos 20 (vinte) expostos na seção 2. O *shell script* deverá ter no mínimo 30 (trinta) linhas de código verificadas pela ferramenta `cloc` (https://github.com/AlDanial/cloc). Cada conceito utilizado na planilha deve ser único e estar acompanhado da aula que o envolve, o trecho de código com número da linha que o utiliza e uma breve explicação. **Os conceitos escolhidos devem ser úteis para a resolução do problema.** Por exemplo, se um *script* possui a atribuição `c=$((3*x+1))` na linha 5, a planilha poderia ser preenchida conforme abaixo. Note que um mesmo trecho de código pode ser utilizado em mais de um conceito, mas cada conceito deve ser único. Por exemplo, se seu *script* utilizar expansão de chaves em outra parte do código você não pode usar o conceito 09 novamente. **O formato de colunas da planilha deve ser idêntico ao mostrado a seguir. A descrição deve deixar claro qual parte do conceito foi utilizado no trecho de código.**

#	Conceito	Aula	Linha	Trecho de Código	Descrição
01	10	Parâmetros e variáveis	5	`c=...`	Variável `c` definida pelo usuário.
	...	...	...	...	...
10	09	Expansões	5	`$((3*x +1))`	Expansão aritmética com `$(())`.

Na terceira etapa, serão realizadas apresentações conforme calendário a seguir. **As apresentações seguirão a ordem em que os problemas foram enviados para a lista de discussão, ou seja, os primeiros a enviarem serão os primeiros a apresentarem.** Você deve explicar o problema resolvido, mostrar o script desenvolvido com sua execução se possível e os conceitos escolhidos com explicação. As apresentações devem ter no máximo 20 minutos.

Data	Aluno(a)
04/12/2025	
09/12/2025	
11/12/2025	
16/12/2025	
18/12/2025	

4. Avaliação

O projeto será avaliado em 30 (trinta) pontos, sendo 2 (dois) pontos para o problema enviado para a lista de discussão, 18 (dezoito) pontos para o relatório e 10 (dez) pontos para a apresentação. A cada dia de atraso no envio do problema para a lista de discussão serão descontados 0,25 (zero, vinte e cinco) pontos. A presença nas apresentações é obrigatória, ou seja, caso o(a) aluno(a) falte será descontado 1 (um) ponto para cada dia de apresentação perdida.