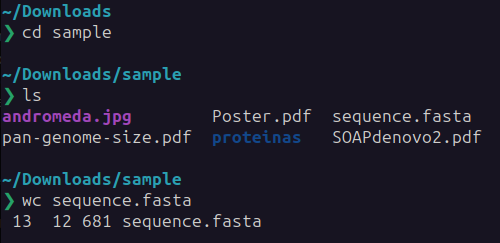
**T01 – SO Linux**

Igor Augusto Reis Gomes – 12011BSI290 – [igor.augusto@ufu.br](mailto:igor.augusto@ufu.br)

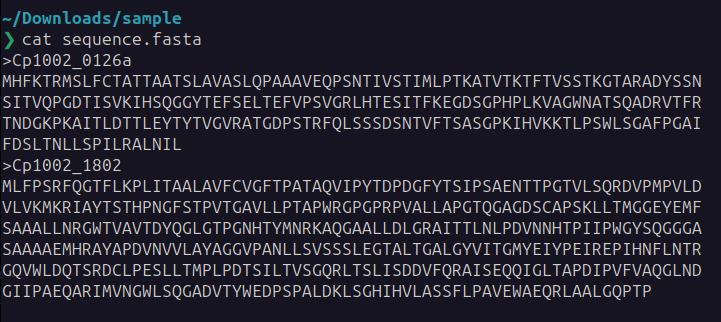
Heitor Guimarães Da Fonseca Filho – 12011BSI203 – [heitor.filho@ufu.br](mailto:heitor.filho@ufu.br)

\*Fizemos apenas os comando avançados dado que já somos usuários de linux (Ubuntu) nativo.

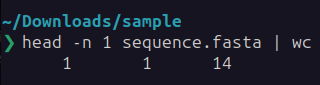
1. Um comando que costuma ser muito útil para saber sobre quantidades de texto de um arquivo é o comando wc (word count). Ele retorna a quantidade símbolos de 'nova linha' (\n), a quantidade de caracteres e de bytes presentes em cada arquivo ou entrada. Acione o comando wc sobre o arquivo sequence.fasta.



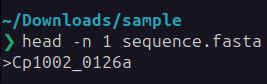
1. Compare o resultado acima com o retornado pelo comando: cat sequence.fasta



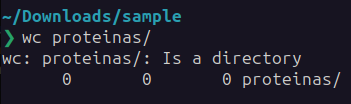
1. O comando wc consegue abrir arquivos por conta própria, mas pode ser que não se queira abrir um arquivo inteiro, mas apenas parte dele. Por exemplo, suponha que você queira ver as estatísticas apenas sobre a primeira linha do arquivo sequence.fasta. Então podemos usar os comandos head em um pipeline com o wc. Veja o resultado deste comando:



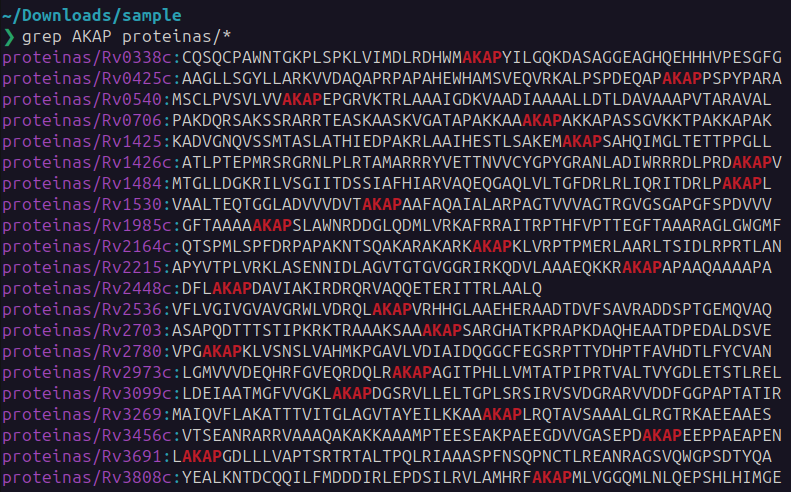
1. Agora digite apenas head -n 1 sequence.fasta e compare com o que foi retornado pelo pipeline.



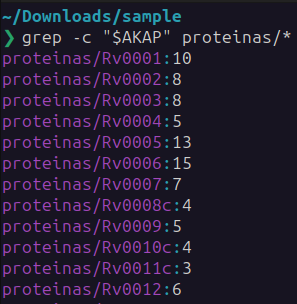
1. Qualquer formato textual gerado pelos comandos pode ser direcionada como entrada para o processamento do comando wc. Por exemplo, como saber quantos arquivos existem dentro do diretório proteinas? (sem acento) Vamos lá, tente, na pior das hipóteses você pode baixar os arquivos novamente e recomeçar!



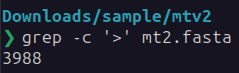
1. O comando grep é bastante útil porque nos permite a consulta por expressões regulares em vários arquivos. Vamos, por exemplo, considerar o exemplo de sala de aula: Um determinado motivo proteico conservado AKAP foi descrito na literatura como importante para a instauração de uma infeção bacteriana. As proteínas do seu organismo de estudo possuem este motivo? Para resolver esse problema basta digitar o comando abaixo e analisar o resultado:



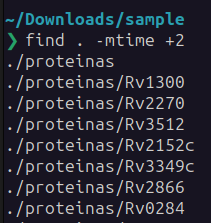
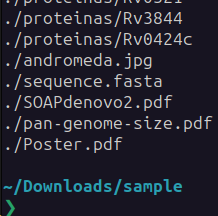
1. O comando grep também possibilita fazer a contagem da quantidade de resultados retornados. Ao invés de você ficar contando na tela a quantidade de motivos retornados (destacados em vermelho) é muito mais prático contá-los com o uso do próprio grep. Acesse o manual do grep e descubra qual parâmetro o configura para fazer a contagem, ao invés de listar o motivo buscado na tela. Em seguida: Use este parâmetro no último comando linux para contar quantos motivos existem.



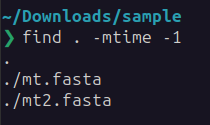
1. O comando grep '>' mt.fasta apenas lista todas as linhas que possuem o sinal de maior. Verifique executando este comando. Perceba que não é isso o que queremos. Dessa forma utilize o parâmetro que você descobriu no item anterior para contar quantas sequencias existem dentro do arquivo. Compare o resultado com o retornado pelo item 2.3.



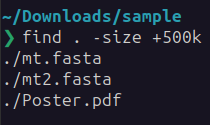
1. Encontre os arquivos modificados a mais de dois dias dentro do diretório sample com o comando find . -mtime +2

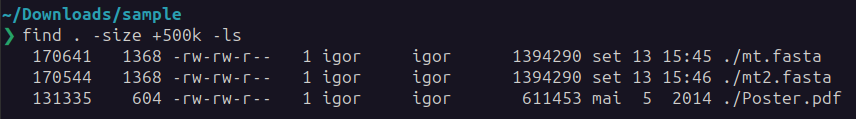
1. Encontre os arquivos modificados a menos de um dia dentro do diretório sample com o comando find . -mtime –1



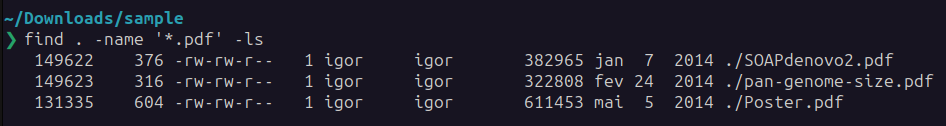
1. Encontre os arquivos filhos que possuem um tamanho maior que 500 kilobytes com o comando find . -size +500k



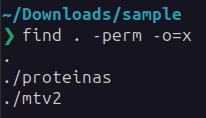
1. Encontre os arquivos filhos que possuem um tamanho maior que 500 kilobytes e liste as propriedades básicas destes arquivos com o comando find . -size +500k –ls



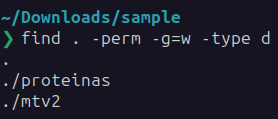
1. Encontre todos os arquivos cujo nome termine com pdf com o comando find . -name '\*.pdf' - ls



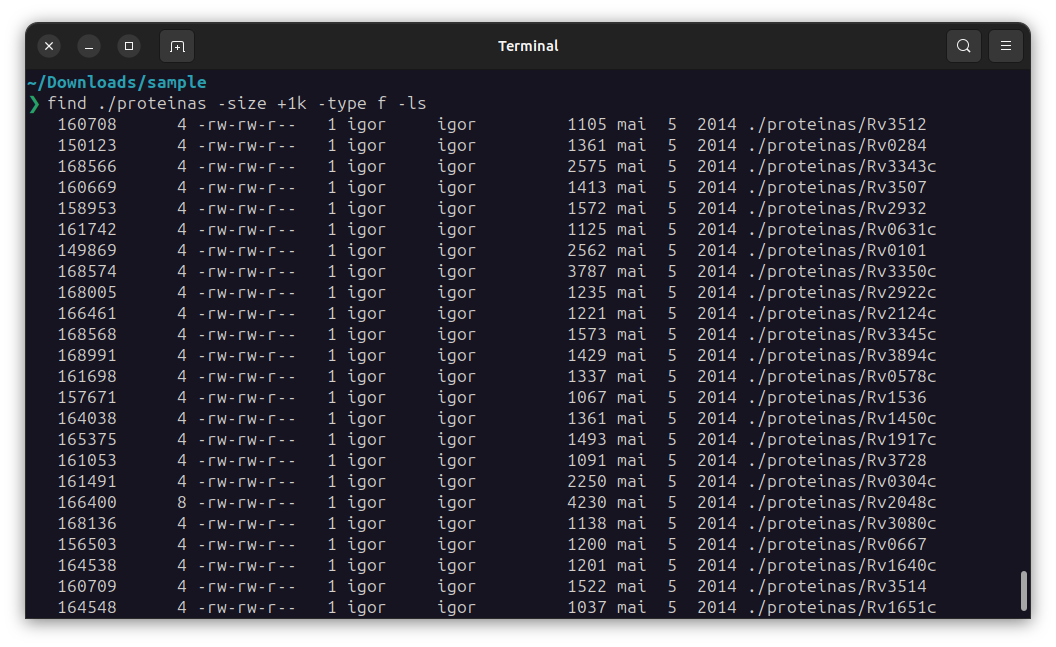
1. Encontre arquivos ou diretórios, a partir do diretório sample, que possuem a permissão de execução para o grupo 'outros'. Lembre-se que existem os grupos de permissões: usuário proprietário, grupo proprietário e outros, que utilizam as letras 'u', 'g' e 'o' para significar cada grupo, respectivamente, sendo que as permissões podem ser r=read, w=write e x=executar. Este comando gera o relatório: find . -perm -o=x



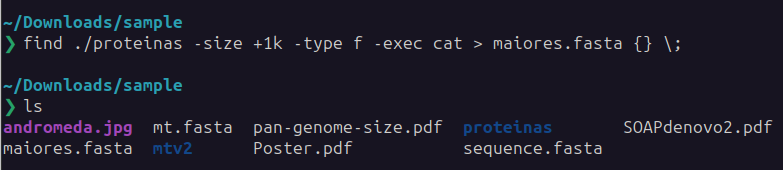
1. Encontre os diretórios a partir da pasta sample que possuem permissão de escrita para o grupo proprietário: find . -perm -g=w -type d



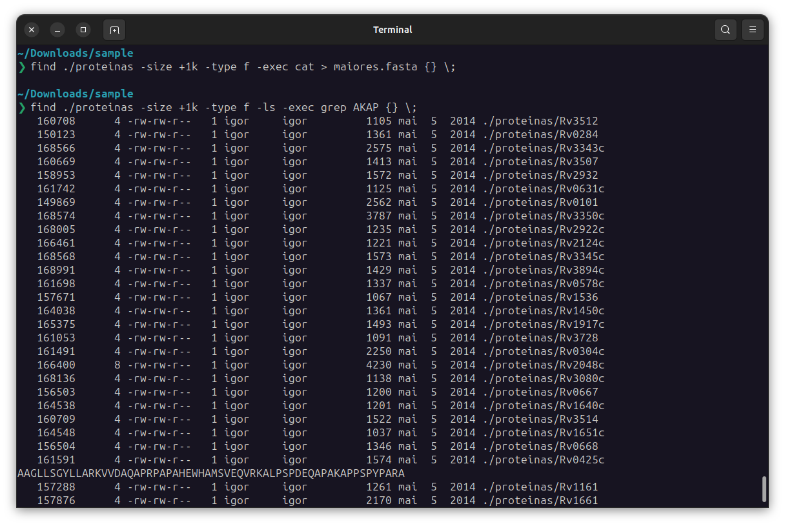
1. Encontre e liste os arquivos de proteínas, da pasta proteinas, apenas com arquivos cujas sequencias de proteínas forem maiores do que 1 kilobyte (1k): find ./proteinas -size +1k -type f –ls



1. Crie um arquivo multi-fasta denominado maiores.fasta, a partir da pasta proteinas, contendo apenas os arquivos com sequencias de proteínas que forem maiores do que 1 kilobyte (1k): find ./proteinas -size +1k -type f -exec cat > maiores.fasta {} \;

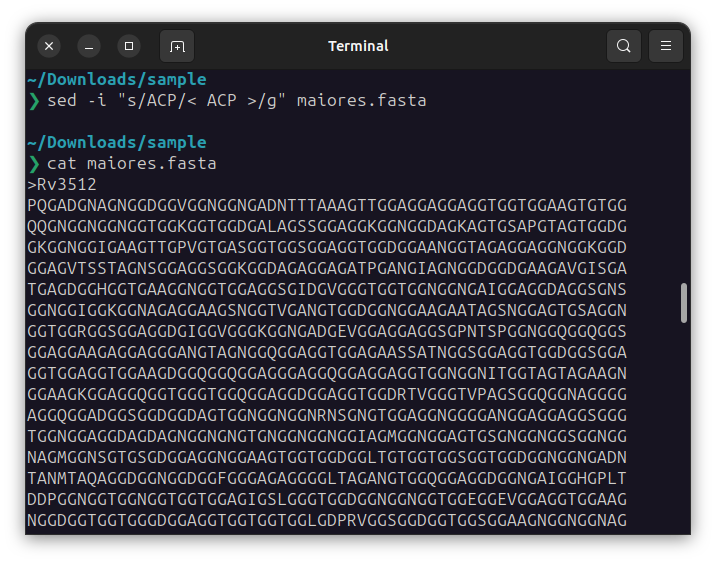
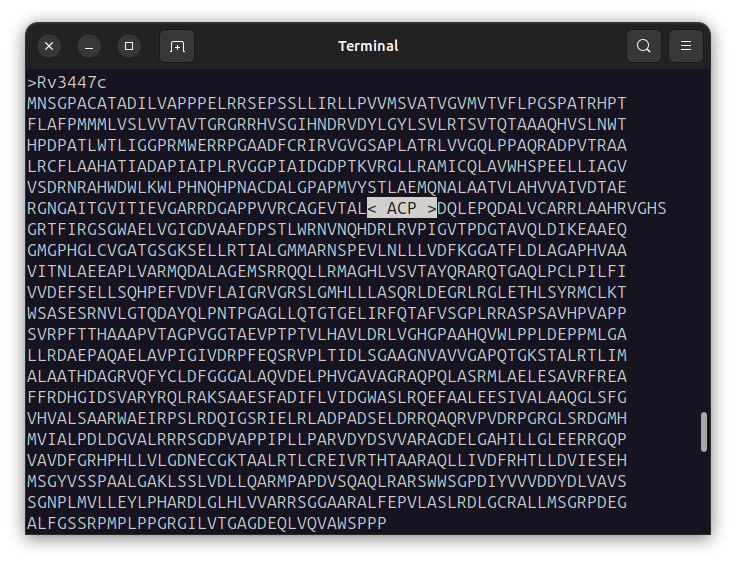


1. Encontre os mesmos arquivos do item anterior, mas ao invés criar um arquivo com as sequencias, execute um comando grep para procurar pelo motivo conservado AKAP: find ./proteinas -size +1k -type f -ls -exec grep AKAP {} \;

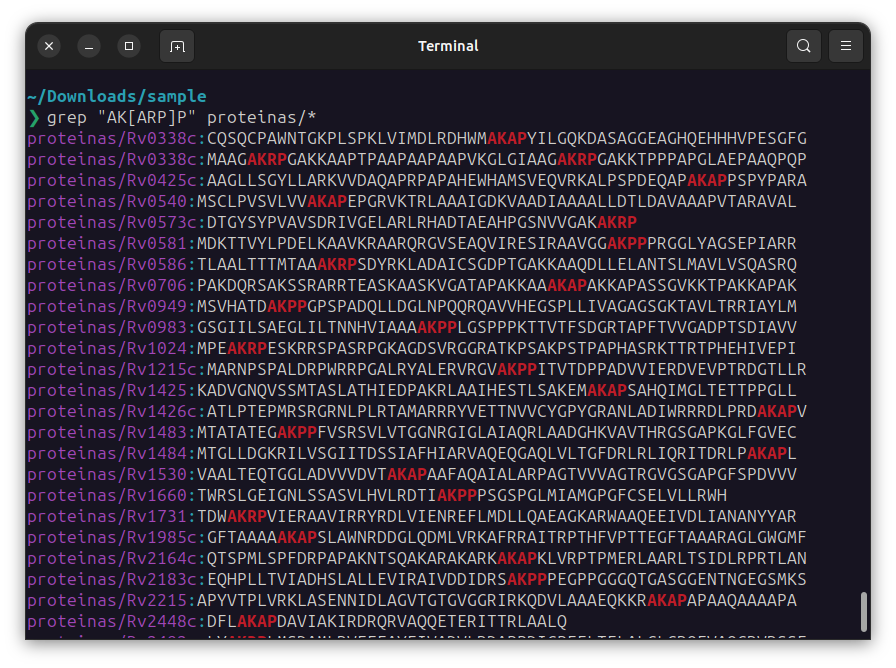


1. O comando sed nos auxilia realizando edições em lote em diversos arquivos. Vejamos um exemplo simples de uso do sed, os exemplos mais interessantes são criados quando conjugamos o sed com expressões regulares. Suponha que uma novo motivo proteico está sendo estudado, o motivo ACP. Deseja-se editar o arquivo maiores.fasta, criado nos item anterior, para delimitar todas as ocorrências de ACP com um tipo de tag, de modo a que este motivo não fique oculto em possíveis análises visuais do arquivo. Para tanto onde houver “ACP” deve ser substituido por “< ACP >”. Execute este comando para realizar tal operação: sed -i "s/ACP/< ACP >/g" maiores.fasta

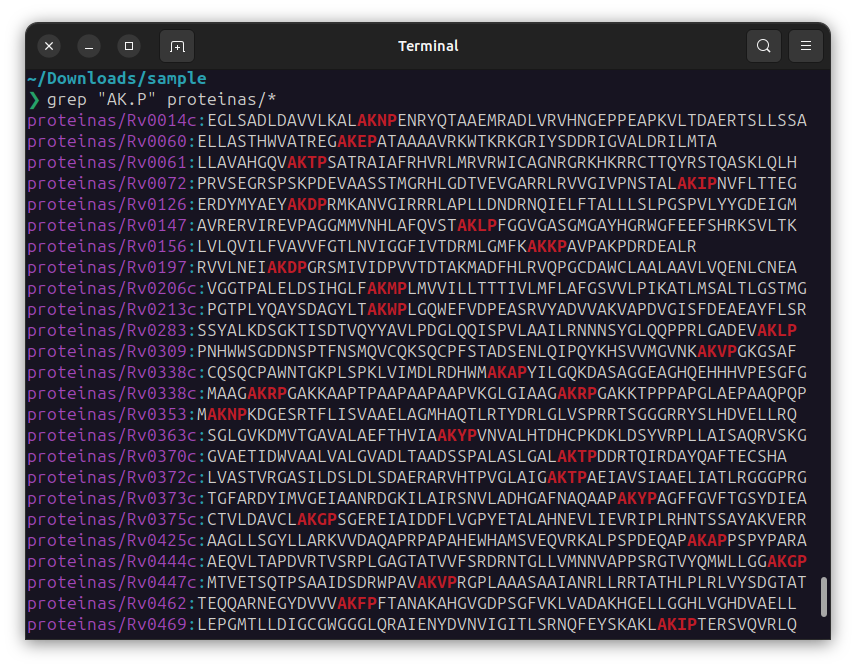
Primeiro print: execução do comando. Segundo: resultado com os < ACP > após usar o cmd “cat”.

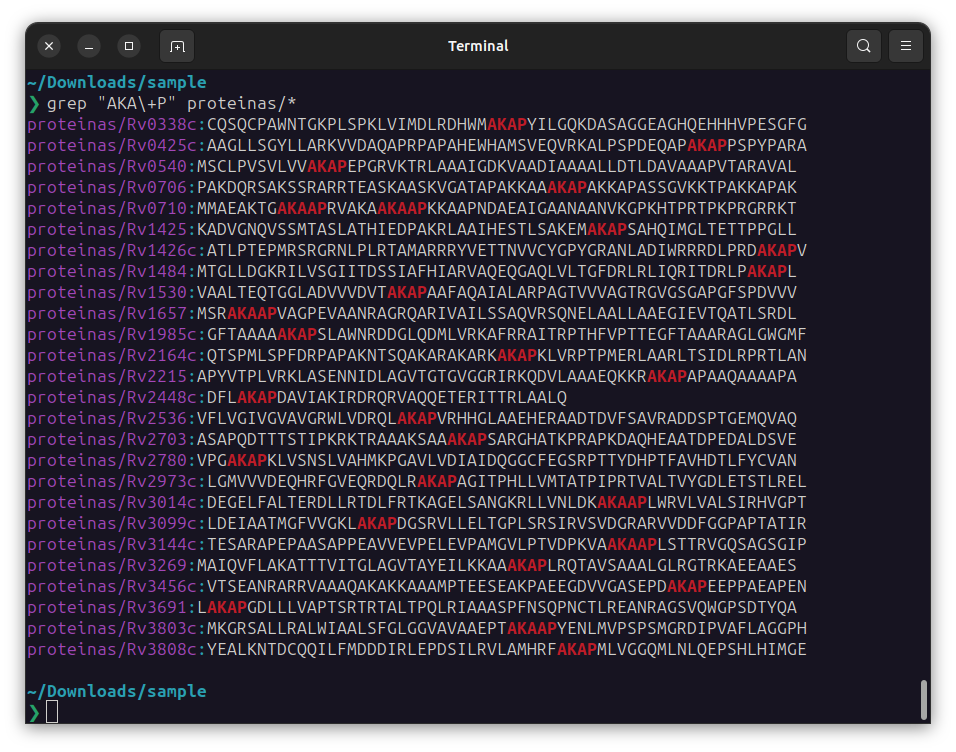
1. Os comandos grep e sed se tornam muito poderosos quando expressões regulares são utilizadas nos parâmetros de busca. Um exemplo simples pode ser a busca de um motivo conservado no qual existe um aminoácido que pode variar entre n letras, digamos n=3 letras. Suponha o motivo AKAP no qual a segunda letra A pode ser substituída pelos aminoácidos R e P. Este comando gera o relatório que precisamos: grep "AK[ARP]P" proteinas/\*



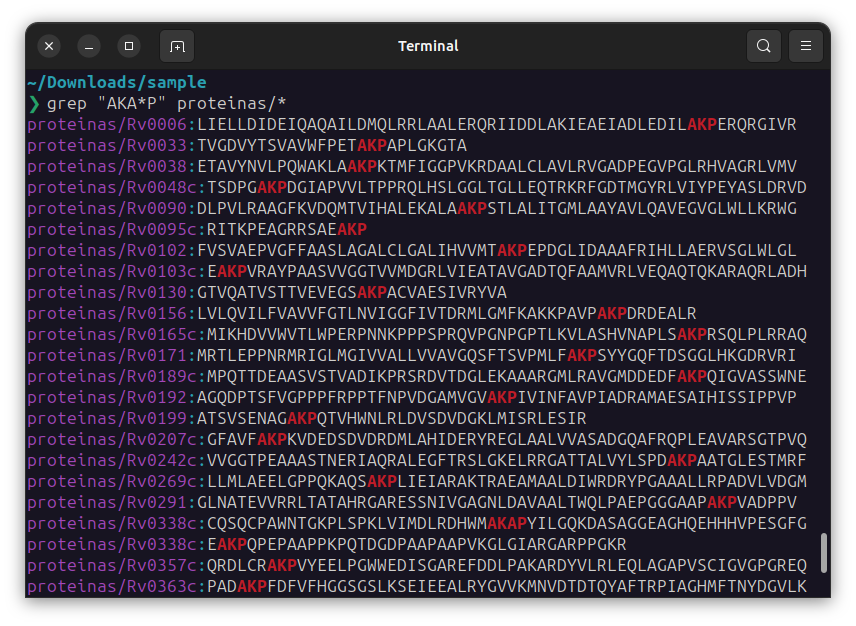
1. Outra variação do comando anterior seria a possibilidade de que qualquer outro aminoácido pudesse entrar no lugar da segunda letra A. Neste caso o ponto '.' vem ao nosso favor: grep "AK.P" proteinas/\*



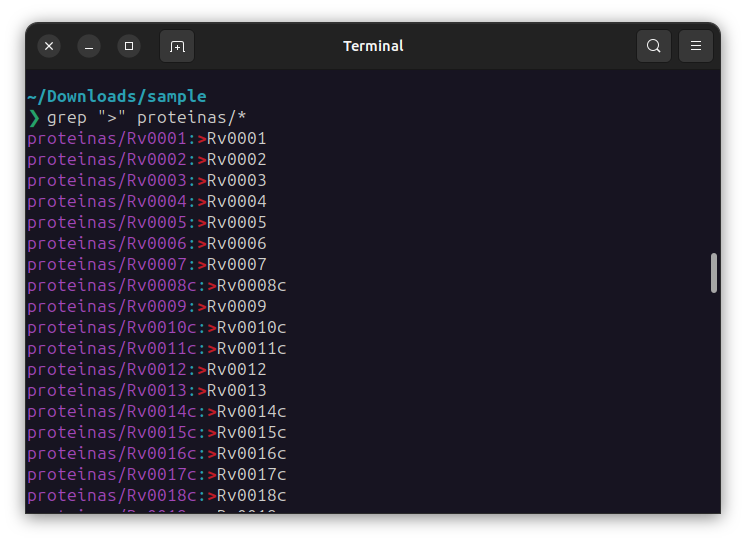
1. Mais uma variação do comando inicial. Suponha que um outro motivo possua uma ou mais letras A no local onde ocorre a segunda letra A. Neste caso o operador de repetição de ER's resolve a nossa consulta. Nessa caso o operador '+' precisa ser precedido por uma contrabarra: grep "AKA\+P" proteinas/\*



1. Uma variação do último comando. Suponha que no lugar do segundo A possa existir zero ou mais ocorrências de A: grep "AKA\*P" proteinas/\*



1. Liste os cabeçalhos dos arquivos presentes no diretório proteinas com o comando: grep ">" proteinas/\*



1. Modifique o cabeçalho de todos os arquivos dentro da pasta proteinas para incluir o prefixo Mt assim adaptar os arquivos para utilização com o tal programa: sed -i "s/>\(Rv[0- 9]\+c\?\)/>Mt\_\1/g" proteinas/\*

