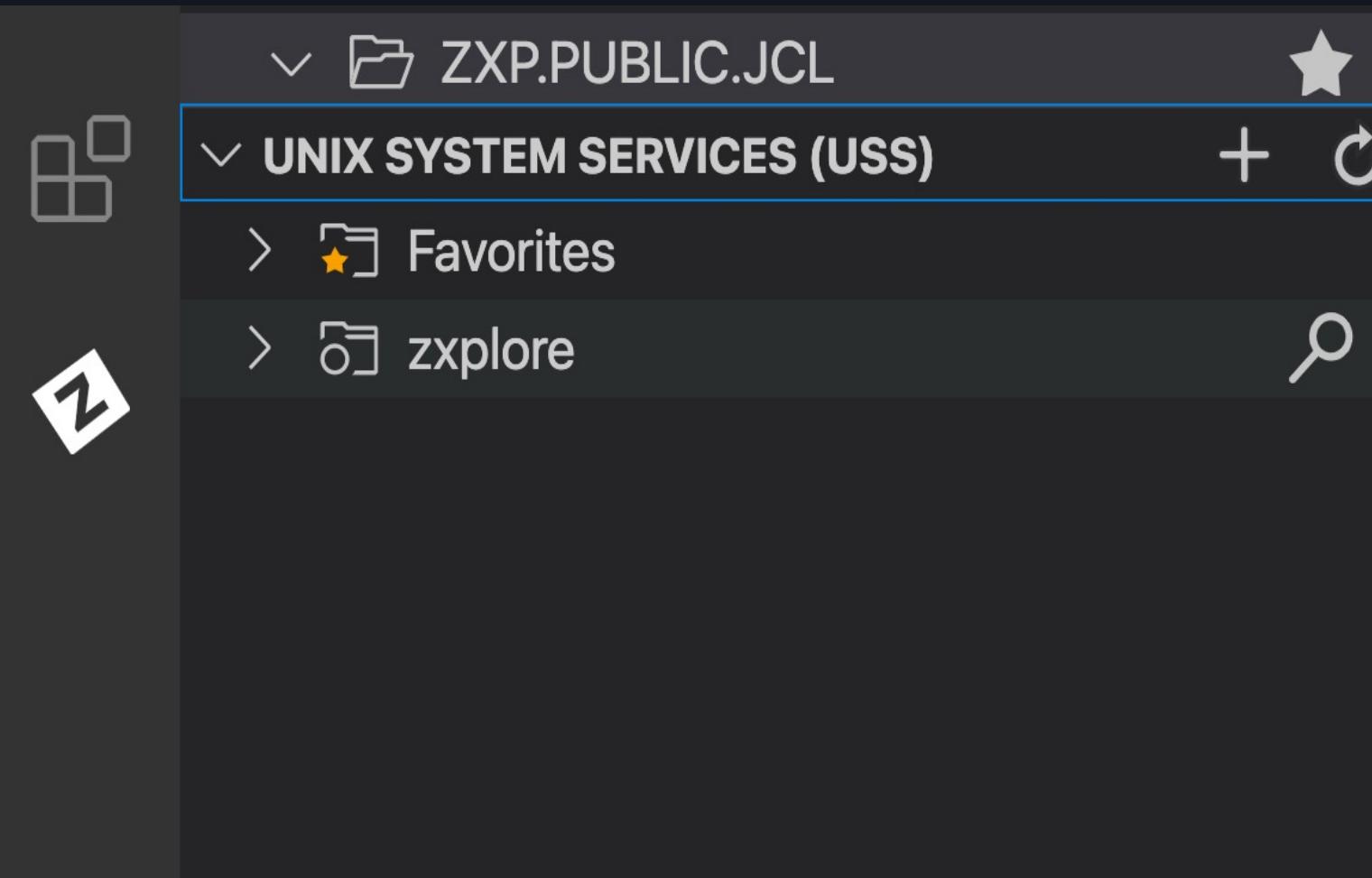


É UNIX. EU SEI DISSO

Sobre scripts, execuções e chmods

12 etapas

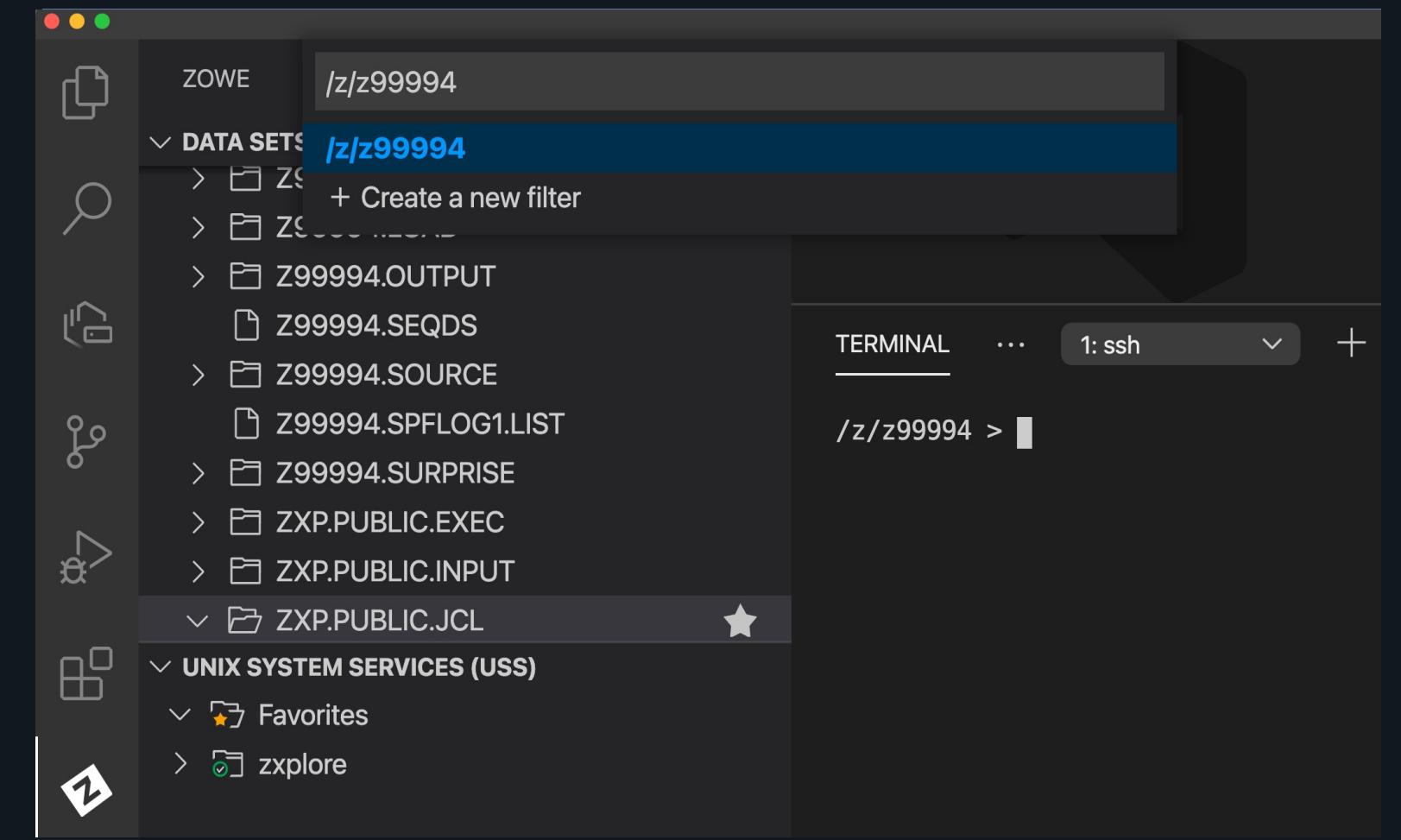
90 minutos



1. ABRA A VISUALIZAÇÃO DO USS

Procure na seção Unix System Service (USS) do VS Code. Se você não mudou o local padrão, ele deve estar no lado esquerdo, entre Conjuntos de dados e Tarefas.

Ele também deve ter um perfil associado (o mesmo usado para os outros dois). No exemplo acima, ele é chamado de **zxplore**. Caso contrário, clique no botão + à direita da barra do UNIX SYSTEM SERVICES (USS) para incluí-lo.



2. CORRIJA SEU FILTRO

Clique no botão *Filtro* (lupa) à direita do perfil do USS no VS Code. Configure para o diretório inicial.

Para relembrar seu diretório pessoal, basta verificar a saída do comando **pwd**, que deve ser semelhante a **/z/zxxxxx** (porém, evidentemente, com os valores do seu próprio ID de usuário).

Dois detalhes importantes:

1. Certifique-se de usar letras minúsculas aqui.
2. Ao efetuar SSH, se ocorrer um “Erro ao recuperar a resposta de uss-file-list”, é possível corrigir digitando o seguinte no terminal: **/z/public/fixfiles.sh**

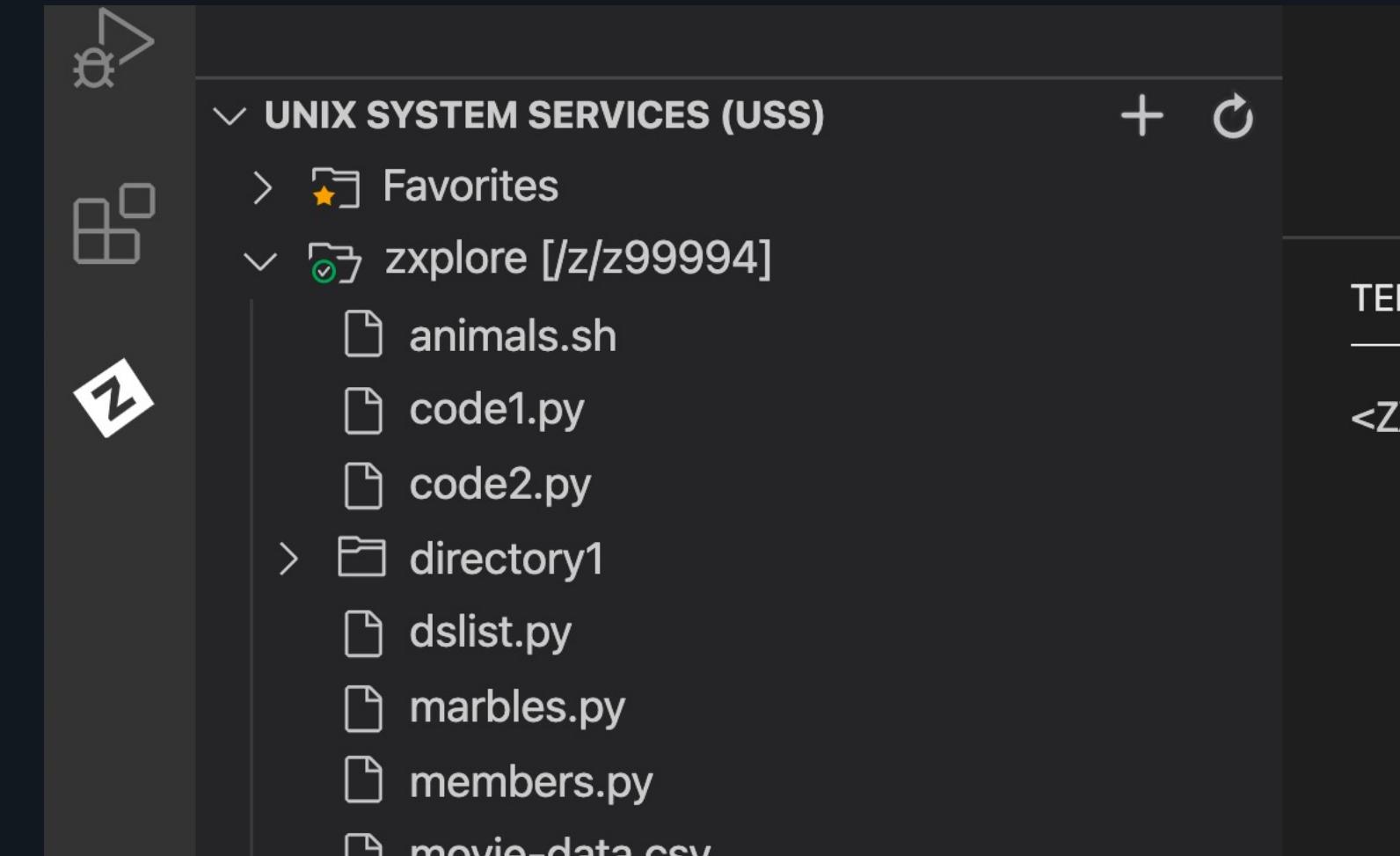
O DESAFIO

Há maneiras de editar arquivos pelo terminal, mas a maioria das pessoas acha muito mais fácil e intuitivo usar o editor de texto integrado ao VS Code.

Para este desafio, vamos analisar o USS com o VS Code, trabalhar em um script de shell e fazê-lo funcionar corretamente para marcar tudo como concluído.

ANTES DE COMEÇAR

Antes de começar, é necessário ter configurado o VS Code e concluído o primeiro desafio do USS em Fundamentos. Fora isso, nada mais é exigido e tudo ocorrerá no USS.



3. ANALISE O CONTEÚDO

Algo parece familiar? É possível usar o VS Code para fazer alguns trabalhos básicos de arquivos e pastas por meio da interface do USS no z/OS, mas o terminal ainda será necessário para emitir comandos e executar scripts, por exemplo. Explore um pouco e note que tudo o que você viu antes está presente, mas de maneira um pouco diferente.

Deseja excluir algo? Confira a caixa azul clara no canto inferior esquerdo da página 3 do USS para obter algumas dicas sobre como excluir arquivos e diretórios. **Deseja renomear algo?**

Clique com o botão direito em um arquivo ou diretório e selecione Renomear. No entanto, recomenda-se não renomear itens, a menos que haja instruções claras para fazê-lo.

```
<ZXP> ssh z99994@204.90.115.200
z99994@204.90.115.200's password:
/z/z99994 > uss-setup
Hi and welcome the ZXPLORE UNIX System Service Challenge!
/z/z99994 >
```

4. UMA LEMBRANÇA PARA VOCÊ

Antes de começar o desafio do USS2, vamos relembrar o que aprendemos.

Primeiro, certifique-se de que o terminal esteja aberto, conectado por meio de SSH e configurado corretamente. Se você não lembrar como fazer isso ou seu diretório padrão estiver vazio, confira a captura de tela acima para encontrar a conexão e *uss-setup* dos fundamentos do USS.

Você sabe como criar um diretório (**mkdir**), criar um arquivo (**touch**), mudar os diretórios (**cd**), analisar o conteúdo (**ls**) e imprimir o diretório de trabalho (**pwd**).

O QUE ESTOU VENDO?

Ao olhar um diretório (ou pasta, se você é fã do Windows), é preciso ter uma ferramenta útil para identificar o conteúdo dele e o que é possível fazer com esse conteúdo (entradas). O comando *ls*, em sua forma mais simples, mostrará os nomes dos arquivos (não ocultos) no diretório, mas há algumas opções comumente usadas para mostrar mais detalhes (confira **man ls** para ver mais):

- -l -- mostra as informações do proprietário, além da data e hora da última atualização e das permissões
- -a -- mostra todos os arquivos, incluindo os ocultos (geralmente, os nomes de arquivos que começam com "." não são listados no Unix)
- -t -- classifica as entradas do diretório pelo horário da última modificação
- -r -- classifica a listagem de entradas do diretório ao contrário
- -F – inclui um código de caractere após o nome da entrada para mostrar se o arquivo é "especial"

```
/z/z99994 > cd ~
/z/z99994 > ls
USSmovies      directory1
animals.sh     dslist.py
code1.py       marbles.py
code2.py       members.py
/z/z99994 > cd USSmovies
/z/z99994/USSmovies > ls
Coco          Finding Nemo Up
/z/z99994/USSmovies >
```

5. USAR TODOS OS ELEMENTOS

Pratique criar um diretório e colocar arquivos nele.

Volte para o diretório inicial no terminal (/z/zxxxxx) e crie um diretório chamado **USSmovies**.

Coloque três arquivos em USSmovies nomeados de acordo com seus filmes favoritos.

Lembre-se de colocar o nome do arquivo entre aspas se ele tiver espaços, por exemplo: touch "mac and cheese".

```
/z/z99994 > ls
USSmovies      directory1
animals.sh     dslist.py
code1.py       marbles.py
code2.py       members.py
/z/z99994 >
/z/z99994 > ls
animals.sh     directory1
code1.py       dslist.py
code2.py       marbles.py
members.py     movie-data.csv
scramble.sh    secret.txt
test          ussout.txt
/z/z99994 >
```

6. LIMPE TUDO

Se isso não é familiar, continue praticando. Continue incluindo arquivos no seu diretório pessoal até se familiarizar com os atalhos e a estrutura.

Se o diretório e os arquivos foram criados com sucesso, agora é hora de excluí-los. Lembra como fazer isso?

Dica: consulte a caixa azul na página 3 do desafio de fundamentos do USS para lembrar.

Depois de remover com sucesso as informações que você acabou de criar, você não verá mais **USSmovies** no diretório (veja a captura de tela acima).

Você está pronto para seguir para a **Etapa 7**. Aumentamos sua independência para que você lide com o UNIX por conta própria.

```

> .vscode > extensions > zowe.vscode-extension
1  #!/bin/sh
2
3  # HELPFUL WORDS FOR CONTESTANT
4  # This script takes three files
5  # and then produces some output
6  # how the script is invoked.
7  #
8  # FILE1, FILE2, and FILE3 should
9  # and the output directory should
10 # The files need to each have
11 # The script needs to be called
12 #
13 FILE1=animal1
14 FILE2=animal2
15 FILE3=animal3

```

7. ABRA animals.sh

Procure no seu diretório pessoal por **animals.sh**. Ele deve ter sido copiado nos fundamentos do USS, quando você executou o script **uss-setup**.

Se não encontrá-lo, copie-o de **/z/public** ou execute o script **uss-setup** novamente, conforme descrito no desafio anterior dos fundamentos do USS.

Este é um script, assim como o outro, mas não vamos somente executá-lo, mas também corrigi-lo para que ele funcione.

"POR QUE O MUNDO PRECISA DE OUTRA LINGUAGEM DE SCRIPT?"

Um número crescente de programas em execução no Z veio do Linux ou de outras plataformas UNIX, nos quais shell scripts são usados para executar etapas de configuração de maneira simples e transparente, como verificar se há espaço em disco suficiente, criar diretórios de destino e garantir que o software de pré-requisito seja instalado.

Uma das razões pelas quais as pessoas realmente gostam de scripts no UNIX é que os comandos são os mesmos usados ao digitar em um shell no modo interativo, mas com o benefício adicional da inclusão de instruções "IF" e "FOR". O sh shell no USS é bastante limitado, mas é possível usar o bash shell mais completamente posteriormente, caso necessário.

```

echo "Checking for the files and folders"
#If there's three files that aren't empty (-s) AND (&&) the directory exist (-d), then do this
if [ -s "$FILE1" ] && [ -s "$FILE2" ] && [ -s "$FILE3" ] && [ -d "$DIRECTORY1" ]; then
    echo "Everything looks good"

#Copy the three files into that directory
cp $FILE1 $FILE2 $FILE3 $DIRECTORY1
echo "Files have been copied!"

#Produce this output and put it in the "message" file
echo "We're extremely happy to have $USERNAME on the system" > "$DIRECTORY1"/message
MODELTYPE=$(uname -m)
OS=$(uname -I)
echo "The operating system is $OS running on a model type $MODELTYPE" >> "$DIRECTORY1"/message
echo "" >> "$DIRECTORY1"/message
echo "If $USERNAME looks out the window, they might say:" >> "$DIRECTORY1"/message

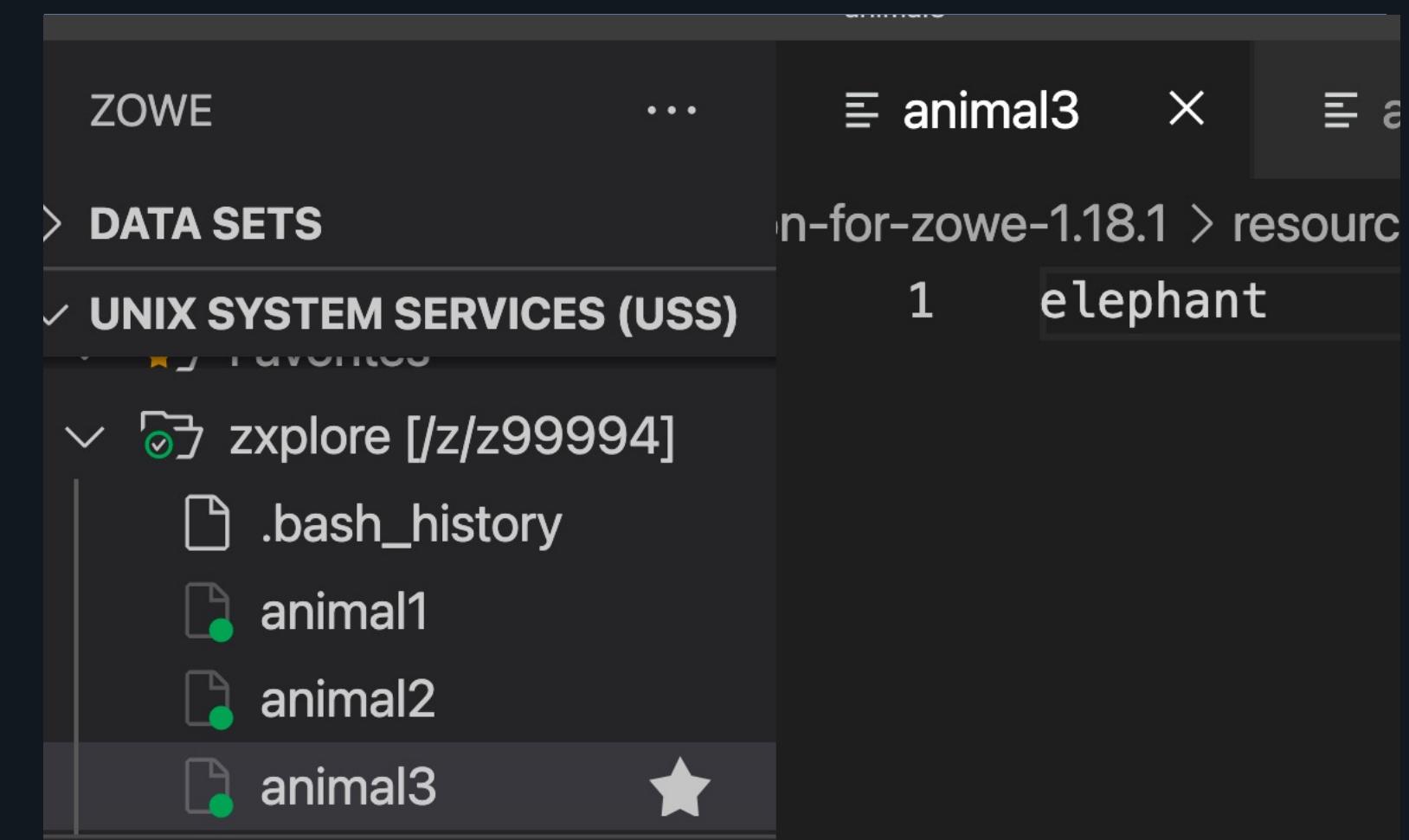
#For every file (there should be three), do this
for file in "$DIRECTORY1"/animal*
do
    CURRENT_ANIMAL=$(cat $file)
    echo "Why hello there, $CURRENT_ANIMAL" >> "$DIRECTORY1"/message
done

```

8. O QUE ESTÁ ACONTECENDO?

Abra o shell script e analise. Há grande parte do script em uma instrução "IF", que só será executada corretamente caso determinadas condições sejam atendidas.

Leia os comentários para obter informações adicionais e tente descobrir o que está acontecendo com o script antes de prosseguir.



9. CORRIJA A ENTRADA

O script tem muitas pistas sobre o que é preciso executar. Crie três arquivos (**animal 1**, **animal 2** e **animal3**) e um diretório chamado **uss2output**.

Use o VS Code para abrir esses arquivos e insira uma única linha em cada um deles, nomeando um animal que você provavelmente verá em sua área. Não pense muito, mas certifique-se de inserir algo, caso contrário, o script não funcionará. Lembre-se de concluir cada linha pressionando Enter ou retorno para que ela seja uma linha, não apenas uma palavra (**cat animal[123]** deve mostrar três linhas).

Observe a captura de tela acima. O arquivo 'animal1' precisa ter um animal nomeado nele. O mesmo é aplicado a animal2 e animal3.

Ao trabalhar entre o terminal e a GUI, pode haver momentos em que as visualizações se tornem inconsistentes. Lembre-se de que pode ser necessário recarregar a visualização do VS Code para obter as atualizações mais recentes.

```

TERMINAL ... 1: ssh + 

/z/z99994 > ls -l animals.sh
-rw-r--r-- 1 Z99994 IPGROUP
/z/z99994 > chmod +x animals.sh
/z/z99994 > ls -l animals.sh
-rwxr-xr-x 1 Z99994 IPGROUP
/z/z99994 > cat animals.sh

```

10. TORNE-O EXECUTÁVEL

Volte para a sessão do terminal (ssh) e efetue **ls -l** em **animals.sh** com **ls -l animals.sh**

Em seguida, torne esse arquivo executável emitindo o comando **chmod +x animals.sh**

Efetue **ls -l animals.sh** novamente e veja se percebe a diferença. Há um **X** incluído nas permissões, informando que ele é executável, ou seja, que ele pode ser executado como um programa. R significa Leitura, W significa Gravação e X significa Execução.

O comando **cat** (abreviação de concatenate), outro comando rápido, exibe por padrão o conteúdo de um arquivo na tela. Use-o para analisar o conteúdo de um arquivo que você não precisa editar.

Por exemplo: **cat animals.sh**



```

TERMINAL ... 1: ssh + 
/z/z99994 > ls -l animals.sh
-rw-r--r-- 1 Z99994 IPGROUP
/z/z99994 > chmod +x animals.sh
/z/z99994 > ls -l animals.sh
-rwxr-xr-x 1 Z99994 IPGROUP
/z/z99994 > ./animals.sh
./animals.sh: 19: Make sure to run this with your name as the first
parameter. Example: ./animals.sh Baron
/z/z99994 >

```

11. EXECUTE O SCRIPT

Digite **./animals.sh**

No UNIX, ao fornecer um caminho para algo, assume-se que desejamos executar essa coisa. Em vez de digitar o caminho completo para o script, é possível usar somente um único ponto e uma barra para indicar “No diretório em que estou agora” e, em seguida, o script que desejamos executar.

O script será executado, mas informará que há um primeiro parâmetro ausente. Siga o exemplo com **seu nome** e tente novamente.

```

PROBLEMS TERMINAL ...
/z/z99994 > cat uss2output/message
You're extremely happy to have built on the system
the operating system is z/OS running on a model type IBM
It just looks out the window, they might say:
Why hello there, cat
Why hello there, dragon
Why hello there, z/OS
They'll say back, "Congratulations on finishing USS Part 2!"
/z/z99994 >

```

12. VERIFIQUE A CONCLUSÃO

Execute o script novamente, mas forneça o primeiro argumento (a parte que vem depois do próprio script) desta vez. O próprio script diz que argumento é esse, além de tudo o que você precisa para fazê-lo funcionar corretamente. Ao final, verifique a saída em **uss2output/message** e você deve ver uma mensagem muito clara parabenizando você pela conclusão da tarefa.

Tudo certo? Ficou feliz? Ótimo. Localize a tarefa **CHKUSS2** em **ZXP.PUBLIC.JCL**, clique com o botão direito nela e selecione **ENVIAR TAREFA** para marcá-la como concluída.

BOM TRABALHO! VAMOS RECAPITULAR

Você escolheu um script, descobriu o que ele faz e o executou. Ao trabalhar no UNIX System Services, você provavelmente fará isso com muita frequência, por isso, é bom que você tenha conseguido descobrir a resposta.

Criamos um script repleto de exemplos de propósito para você, pois é possível emprestar algumas dessas expressões e verificações condicionais mais tarde para suas próprias criações. (dica)

EM SEQUÊNCIA...

Se você já concluiu os desafios do PDS e do JCL2, sua próxima tarefa é começar os desafios do ZOAU. Eles usam tudo o que você aprendeu até agora e reúnem todas essas coisas de maneira única para você mostrar realmente suas habilidades.