[✒️](https://coolsymbol.com/copy/Fountain_Pen_Emoji_Symbol_%E2%9C%92%EF%B8%8F" \t "_blank)**Introduction**

Hello EveryOne, Today I'm starting a 7 Day's Bash Script Challenge organized by [Shubham Londhe](https://hashnode.com/@TrainWithShubham) . Here every day we will dive into Script, If you are interested in this then you also join this challenge, where you will learn new things.

As we talk about our Day 1 challenge we will be learning about the basics of Bash Script, Without any further discussion we will start.

[✒️](https://coolsymbol.com/copy/Fountain_Pen_Emoji_Symbol_%E2%9C%92%EF%B8%8F)**What is Bash Script?**

Hey there! 🎉 Imagine you have a magical fairy assistant 🧚‍♂️ that follows your every command! Well, that's kind of what a Bash script is! 🌟

💻 Bash is like a wizard 🧙‍♂️ who speaks computer language and can perform amazing spells on your computer! With a Bash script, you give the wizard a list of instructions written in a special language (Bash script), and he magically executes them one by one!

📝 The script is like a recipe 🍲 for the wizard. You write down all the steps you want him to follow, like "wave the wand," "mix the potion," and "cast the spell" 🪄 - those are your commands!

🔍 Then, the wizard reads the recipe (your script) line by line, and BOOM! 💥 Your computer starts doing all the cool stuff you asked for! It can create folders, move files, talk to other programs, and so much more!

✨ And the best part? You can reuse the same magical recipe over and over again to make the wizard perform the same spell! It saves you lots of time and effort! 🕰️

💡 So, when you hear about a "Bash script," think of it as a magical spellbook 📚 that lets you control your computer with just a few lines of special wizardry language! 🧙‍♀️🌟

**Task:1 Comments**

📝 Imagine you're writing a secret diary 📔, and you want to add little notes to remind yourself of what's happening in your adventurous journey. Well, that's exactly what comments are in a Bash script! 💬

🌟 Comments are like hidden messages 💌 that you write in your script, meant only for you (and other wizards who read your code). They start with a special symbol, the "hashtag" or "pound sign" (#). It's like casting an invisibility spell on your words, so the computer ignores them completely! 🚫🧙‍♂️

🗝️ So, when you want to remember what a particular part of your script does, or if you want to leave a clue for fellow wizards who might look at your code, you sprinkle these magical hashtags and share your thoughts! It's like leaving breadcrumbs in the enchanted forest of code. 🌳🏰

🔍 Here's the fascinating part: the spells you write in your script are spoken in the language of computers, but comments are your secret language, meant only for humans to understand! 🗣️💻

💭 So, next time you're casting your Bash spells, remember to use the power of comments to keep track of your adventures and guide your fellow wizards through the mystical world of your code! 🧙‍♀️🌌

COPY

#!/bin/bash

# DevOps Engineer: This is a bash script that demonstrates basic concepts of bash scripting.

# Task 1: Comments

# - In this task, we will use comments to explain the script.

# Output a welcome message to the user

echo "Welcome to Day 1 of the Bash Scripting Challenge!"

**Task:2 Echo**

🗣️ Imagine you're a powerful sorcerer with a magical megaphone 📢 that can make your voice echo throughout the entire kingdom! Well, that's exactly what the "echo" command does in a Bash script! 🧙‍♂️💬

💻 When you use the "echo" spell in your script, you're casting a charm that makes your words appear on the screen for all to see! It's like leaving sparkly messages 💌 that everyone can read, including other wizards and even your computer! 🧚‍♀️✨

🌈 Just like a fairy spreading fairy dust, "echo" sprinkles your words on the display, creating a magical connection between you and your computer. You can send greetings, share secret incantations, or display the results of your mystical calculations! 📜🔮

📝 To cast the "echo" spell, you simply write "echo" followed by the words you want to share. Your computer listens carefully to your commands, and when it encounters "echo," it gleefully displays your message on the screen like a shimmering spell coming to life! 🪄✨

💭 So, when you want to communicate with your computer or let others in on your sorcery, just reach for the "echo" spell, and watch as your words fill the realm of code with wonder and enchantment! 🗝️🌌

COPY

#!/bin/bash

# DevOps Engineer: Task 2: Echo

# - In this task, we will use the 'echo' command to print a message.

# Output a welcome message to the user

echo "Welcome to Day 1 of the Bash Scripting Challenge!"

**Task:3 Variables**

🎭 Imagine you're a clever magician 🔮, and you have a magical box 📦 where you can store all sorts of treasures! Well, in the realm of Bash scripts, that magical box is called a "variable"! 🎩

🧙‍♂️ Variables are like containers that hold valuable information, just like your magical box holds precious gems and spells! They have unique names, and you can put different things inside them, like numbers, words, or even whole sentences! 🌈🔢🗣️

🔍 Whenever you need to remember something important during your magical incantations, you can store it in a variable. It's like labeling your potions, so you can easily find them later! ✉️🏷️

📝 And here's the exciting part: you can change the contents of your magical box whenever you like! So, if you need a new gemstone or a different spell ingredient, you can swap it out in a snap! 💎🌟

🤝 Variables are like messengers 🏃‍♀️ between your spells. They carry information from one part of your script to another, helping your spells work together harmoniously! It's like having magical assistants who can fetch and deliver things for you! 📬🧚‍♀️

💡 So, whether you're counting enchanted stars, storing the name of your magical pet dragon, or keeping track of your quest progress, variables are your trusty companions on this mystical coding adventure! 🐉✨

COPY

#!/bin/bash

# DevOps Engineer: Task 3: Variables

# - In this task, we will declare and use variables.

# Declare variables and assign values

name="Purushotam"

age=20

# Output the values of variables

echo "Name: $name"

echo "Age: $age"

**Task 4: Using Variables**

Now that we have declared variables, let's use them to perform a simple task. In this task, we will create a bash script that takes two variables (numbers) as input and prints their sum using those variables.

#!/bin/bash

# DevOps Engineer: Task 4: Using Variables

# - In this task, we will use variables to perform a calculation.

# Get user input for two numbers

read -p "Enter the first number: " num1

read -p "Enter the second number: " num2

# Calculate the sum of the two numbers

sum=$((num1 + num2))

# Output the result

echo "The sum of $num1 and $num2 is: $sum"

**Task 5: Using Built-in Variables**

Bash provides several built-in variables that hold useful information. Your task is to create a bash script that utilizes at least three different built-in variables to display relevant information.

#!/bin/bash

# DevOps Engineer: Task 5: Using Built-in Variables

# - In this task, we will use built-in variables.

# Output the current user

echo "Current user: $USER"

# Output the hostname

echo "Hostname: $HOSTNAME"

# Output the current working directory

echo "Current directory: $PWD"

**Task 6: Wildcards**

🌟 Imagine you have a magical key 🔑 that can unlock countless doors with just one turn. Well, that's precisely what wildcards are in a Bash script! They are like powerful, all-purpose keys that let you unlock and manipulate files and folders in the blink of an eye! 🚪🔓

🗝️ Wildcards are represented by special characters like the "asterisk" (\*) and the "question mark" (?). When you include these mystical symbols in commands, the Bash wizard takes it as a sign to unleash their magic! ✨🧙‍♀️

🌌 The "asterisk" (\*) is like a shooting star 🌠 in the night sky. It matches any number of characters, allowing you to find files with mysterious and ever-changing names. It's like having a shape-shifter at your command! 🌟🧞‍♂️

🌈 On the other hand, the "question mark" (?) is like a treasure map with missing pieces. It stands for just one unknown character, making it perfect for uncovering hidden files with puzzling names! It's like having a crystal ball to predict the unknown! 🔍🔮

🏰 With these wildcards, you can conjure spells like "list all the files that start with 's'," or "delete everything ending with '.txt'." Your commands become a symphony of mystery and discovery! 🎶🎇

💡 So, when you venture into the world of Bash scripts, don't forget to wield the power of wildcards! They are your trusty companions on this enchanting journey, helping you navigate the vast landscapes of files and directories with ease! 🗺️🧭

**Conclusion**

In the enchanting world of Bash scripting, you have embarked on a magical journey filled with powerful spells and captivating adventures! 🧙‍♂️✨

💡 You learned the secret language of Bash, where variables act as mystical containers, holding the essence of your data and secrets close to your heart! 🎁💖

🌟 Command spells allowed you to wield the power of your computer, bending it to your will with every line you wrote! 🚀🖊️

🃏 Wildcards became your all-purpose keys, unlocking doors to hidden files and folders, revealing the mysteries of the digital realm! 🗝️🌌

💬 And amidst your spellbinding code, comments became your whispered confidants, guiding you through your own magical incantations and helping fellow wizards comprehend the secrets within! 📝🤝

🧙‍♀️ Now, armed with this newfound knowledge, you can weave your own captivating Bash scripts, orchestrating symphonies of code that dance and sing in harmony! 🎶🌈

So go forth, young sorcerer, and embrace the beauty of Bash scripting! Let your creativity soar as you conjure up amazing scripts, unlocking the full potential of your computer's magic! 🧚‍♂️🌟🔮

[**Checkout My GitHub Repository for projects:**](https://github.com/PurushotamSharma)

Thanks for reading this blog. If you like this blog and find helpful then share with your friends. Happy Learning.