Introduction to Shell Programming

1. What are environment variables?

a. Do you think some of them have a scope similar to global variables in a programming language (i.e. available to all the users on the system) and local variables (i.e. available only to a specific user who has set them)?

b. How can one find out the environment variables available to all the users on a Linux system?

c. How can one find out the environment variables set specifically for auser? How do you set them?

A) Environment variables are dynamic values stored in the shell's environment. They are used by the operating system and applications to configure behavior (e.g., PATH, HOME, USER)

a. Scope of Environment Variables

Yes, environment variables can be compared to global and local variables:

System-wide (global) environment variables → available to all users (like global variables).

User-specific environment variables → only available to the user who set them (like local variables).

b. Find environment variables available to all users

System-wide environment variables are usually defined in:

- /etc/environment
- /etc/profile
- /etc/bash.bashrc
- Files in /etc/profile.d/

c. Find and set user-specific environment variables

User-specific variables are set in shell startup files in the user's home directory:

- ~/.bashrc
- ~/.profile
- ~/.bash_profile

To **see all current environment variables**

Printenv or env

To **set a variable temporarily**

export MYVAR="Hello"

To **set a variable permanently for a user**

source ~/.bashrc

2. Write a shell program that prints the numbers from 1 to 100 with each line having the number in digits, followed by a space and then the number in words.

For example, the expected output would be:

1 one

2 two

3 three

...

99 ninety nine

100 hundred

```
number_to_words() {
    local num=$1
    local word=""

    # Arrays for word representation of numbers
    local ones=( "" "one" "two" "three" "four" "five" "six" "seven" "eight" "nine" )
    local teens=( "ten" "eleven" "twelve" "thirteen" "fourteen" "fifteen" "sixteen"
"seventeen" "eighteen" "nineteen" )
    local tens=( "" "" "twenty" "thirty" "forty" "fifty" "sixty" "seventy" "eighty"
"ninety" )

    # Handle the number 100
    if [[ $num -eq 100 ]]; then
        word="hundred"
    # Handle numbers from 1 to 9
    elif [[ $num -lt 10 ]]; then
        word="${ones[$num]}"
    # Handle numbers from 10 to 19
    elif [[ $num -ge 10 && $num -lt 20 ]]; then
        # Subtract 10 to get the correct index for the 'teens' array
        word="${teens[num-10]}"
    # Handle numbers from 20 to 99
    else
        # Get the tens digit and the ones digit
        local tens_digit=$((num / 10))
        local ones_digit=$((num % 10))
        word="${tens[$tens_digit]}"
        # If the ones digit is not 0, add a space and the word for the ones digit
        if [[ $ones_digit -ne 0 ]]; then
            word="${word} ${ones[$ones_digit]}"
```

```
        fi
    fi
    echo "$word"
}
# Loop from 1 to 100
for i in $(seq 1 100); do
    word=$(number_to_words $i)
    echo "$i $word"
done
```

RUN: chmod +x num_words.sh

./num_words.sh

3. How many unique processes are currently running on your Linux system? Hint: You may explore the '-o cmd' option of ps command in conjunction with sort, uniq and wc. Can you try another solution with the help of the cut command?

A) ps -eo cmd | sort | uniq | wc -l OR ps -eo cmd | cut -d " " -f1 | sort | uniq | wc -l

4. Write a shell script that takes 2 integers as command line arguments and prints their sum. Does your script work if one or both of the integers are negative numbers or zero? If not, modify your program or the way you run the script to make it work.

A) #!/bin/bash

if [$# -ne 2]; then

   echo "Usage: $0 num1 num2"

   exit 1

fi

sum=$(($1 + $2))

echo "Sum: $sum"

Run: ./add.sh 10 -5

Output: Sum: 5

5. Explore the grep command. Can you find an alternative for wc -l, using grep?

A) Count line in a file

grep -c "" filename

Count lines from a command's output:

ls | grep -c ""

6. What commands (or combination of commands) can be used to pipe the output of the program in Question 2 so that only the lines containing 2-digit numbers (i.e., 10 to 99) are printed?

A) ./num_words.sh | grep "^[1-9][0-9] "

Github: GitHub - siva2555/Matrimorphosis: Assignments