Report of LAB02

کیان پورآذر ۴۰۱۳۱۴۰۳

امیرحسین متقیان ۴۰۱۳۱۰۴۳

سوال اول:

```
#!/bin/bash

read -p "Enter the temperature: " temp

if (( temp < 0 )); then
    echo "The weather is freezing"
elif (( temp >= 0 && temp <= 30 )); then
    echo "The weather is cool"
else
    echo "The weather is hot"
fi
```

این برنامه عددي را به عنوان ورودي آرگومان از کاربر دریافت می کند. در صورتی که عدد کوچک تر از صفر بود، عبارت ' The freezing is weather ' و در صورتی که بین صفر و 30 درجه بود، عبارت' The cool is weather ' و در صورتی که دما بالاي 30 درجه بود، عبارت 'hot is weather The‘ را در ترمینال چاپ می کند.

سوال دوم:

```
#!/bin/bash

read -p "Enter the first number: " num1
read -p "Enter the second number: " num2
read -p "Enter an operator (+, -, *, /): " operator

if ! [[ "$num1" =~ ^-?[0-9]+([.][0-9]+)?$ ]] || ! [[ "$num2" =~ ^-?[0-9]+([.][0-9]+)?$
]]; then
    echo "Error: One of the inputs is not a valid number."
    exit 1
fi

case "$operator" in
    +)
        result=$(echo "$num1 + $num2" | bc)
```

```
        echo "Result: $result"
        ;;
    -)
        result=$(echo "$num1 - $num2" | bc)
        echo "Result: $result"
        ;;
    '*')
        result=$(echo "$num1 * $num2" | bc)
        echo "Result: $result"
        ;;
    /)
        if [[ "$num2" == 0 ]]; then
            echo "Error: Division by zero is not allowed."
        else
            result=$(echo "scale=2; $num1 / $num2" | bc)
            echo "Result: $result"
        fi
        ;;
    *)
        echo "Error: Invalid operator. Please use one of +, -, *, /."
        ;;
esac
```

این برنامه به عنوان عدد اول، عدد دوم و علامت ریاضی علامتی که میخواهیم بین این دو انجام بشود را به عنوان ورودي دريافت و عنوان خروجی نتیجه را به ما نشان می دهد، در صورتی که ورودیها نامعتبر بود یا در انجام عملیات دچار مشکل بشویم نیز خروجی مرتبط به آن نمایش داده می شود.

سوال سوم:

```
read -p "Enter the temperature in Celsius: " temp_celsius

if ! [[ "$temp_celsius" =~ ^-?[0-9]+([.][0-9]+)?$ ]]; then
    echo "Error: Please enter a valid number for the temperature."
    exit 1
fi

temp_fahrenheit=$(echo "scale=2; ($temp_celsius * 9/5) + 32" | bc)

if (( $(echo "$temp_celsius < 0" | bc -l) )); then
    echo "The weather is freezing"
elif (( $(echo "$temp_celsius >= 0" | bc -l) && $(echo "$temp_celsius <= 30" | bc -l)
)); then
    echo "The weather is cool"
else
    echo "The weather is hot"
fi
```

```
echo "The temperature in Fahrenheit is: $temp_fahrenheit °F"
```

همانند سوال اول عمل میکند با این تفاوت که دمای وارده به سانتی‌گراد را به فارانهایت تبدیل و چاپ می کند.

سوال چهارم:

```
#!/bin/bash

read -p "Enter a number: " number

if ! [[ "$number" =~ ^[0-9]+$ ]]; then
    echo "Error: Please enter a valid positive integer."
    exit 1
fi

reversed_number=$(echo "$number" | rev)

echo "The reversed number is: $reversed_number"
```

این برنامه اعداد را آیینه میکند به عبارتی دیگر مثلا ما به آن عدد ۶۵۹ داده ایم و در خروجی عدد ۹۵۶ چاپ میشود.

سوال امتیازی استاد:

```
#!/bin/bash

echo "Press [CTRL+C] to stop.."
while true; do
    mkdir NewFolder
    echo "New Folder Created
successfully :)"
    cd NewFolder
    sleep 1
done
```

یان برنامه با استفاده از loop while فولدر های تودرتو میسازد.

سوال ۲ امتیازی استاد:

```bash
#!/bin/bash


LOG_FILE="cpu_load_log.txt"


echo "I'm recording your CPU Load. (Press Ctrl+C for STOPPING!)"
while true; do
    CURRENT_TIME=$(date +"%Y-%m-%d %H:%M:%S")

    CPU_LOAD=$(top -l 1 | grep "CPU usage" | awk '{print $3}')

    echo "$CURRENT_TIME - CPU Load: $CPU_LOAD" >> "$LOG_FILE"

    sleep 1
done
```

این برنامه لوود CPU را با استفاده از top و grep میگیرد و همراه با تایم دقیق آن را داخل فایل cpu_load_log.txt سیو میکند.

سوال ۳ امتیازی استاد:

سطح دسترسی Bash به سطح دسترسی کاربری که اسکریپت را اجرا می‌کند، بستگی دارد .Bash به خودی خود محدودیتی برای دسترسی‌ها ندارد؛ در عوض، دسترسی به فایل‌ها، فولدرها و منابع سیستم به سطح دسترسی کاربر محدود می‌شود. در سیستم‌های) Unix-like مانند لینوکس و(macOS ، کاربران می‌توانند با دو نوع سطح دسترسی معمول اسکریپت‌های Bash را اجرا کنند:

۱.سطح دسترسی کاربر عادی

- به فایل‌ها و فولدرهایی که متعلق به کاربر است یا کاربر دسترسی مناسبی به آنها دارد، دسترسی خواهد داشت.
- محدودیت‌هایی در دسترسی به فایل‌های سیستمی و فولدرهایی که نیاز به سطح دسترسی ریشه دارند، وجود دارد. به عنوان مثال، کاربر نمی‌تواند به فایل‌های حساس در مسیر `/etc` یا `/root` دسترسی داشته باشد.

۲.سطح دسترسی ریشه(Root)

- اگر اسکریپت Bash با کاربر ریشه (یا از طریق `sudo`) اجرا شود، دسترسی کاملی به تمام فایل‌ها، فولدرها و منابع سیستم خواهد داشت.
- می‌تواند به طور مستقیم فایل‌های سیستمی را تغییر دهد، خدمات را متوقف و شروع کند و به منابع حساس دسترسی پیدا کند.
- اجرای اسکریپت‌ها با سطح دسترسی ریشه باید با احتیاط انجام شود، زیرا ممکن است تغییرات غیرقابل بازگشتی در سیستم ایجاد کند.

برای اجرای اسکریپت Bash با سطح دسترسی ریشه، معمولاً از دستور `sudo` به شکل زیر استفاده می‌شود:

```
sudo ./script_name.sh
```

بنابراین، سطح دسترسی Bash تابعی از دسترسی‌های کاربری است که اسکریپت را اجرا می‌کند و Bash به خودی خود محدودیتی ندارد.