Assignment:3

Task:2

2: Debugging Scripts

Objective: Write a buggy shell script and use debugging to identify and fix the issue.

Certainly! Here's a shell script with a deliberate bug, and you can use debugging to identify and fix the issue:

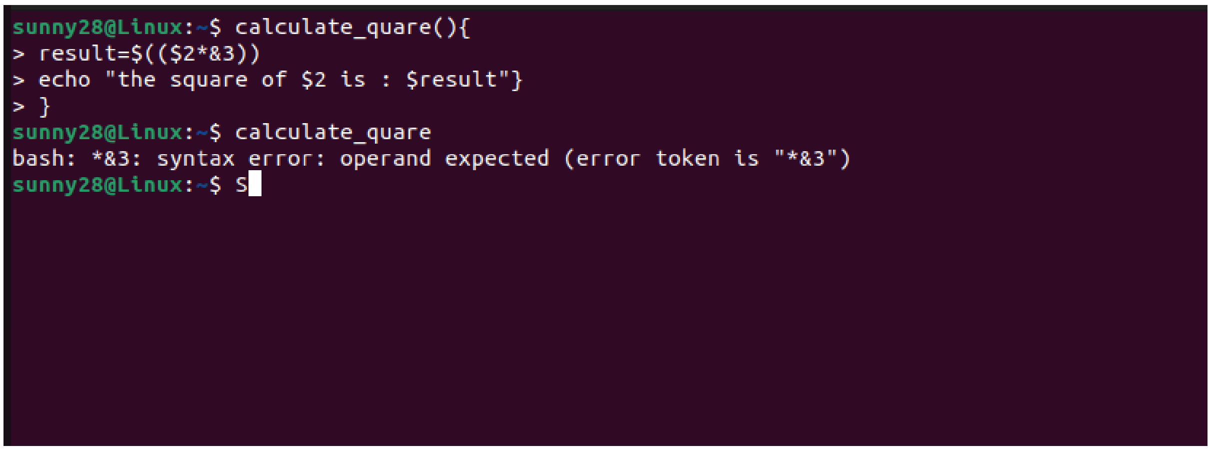
calculate_quare() {

result=$(($1 * $1))

echo "The square of $1 is: $result"}

}

calculate_quare



```
sunny28@Linux:~$ calculate_quare(){
> result=$(($2*&3))
> echo "the square of $2 is : $result"}
> }
sunny28@Linux:~$ calculate_quare
bash: *&3: syntax error: operand expected (error token is "*&3")
sunny28@Linux:~$ S
```

Certainly! there's a syntax error in this script. Let's debug it using the set -x and set +x commands to trace the execution and identify the issue. Also, note that there's an extra curly brace at the end of your function.

calculate_square() {

result=$(($1 * $1))

echo "The square of $1 is: $result"

```
}

# Enable debugging
set -x

# Call the function to calculate the square
calculate_square 5

# Disable debugging
set +x
```

Now, follow these steps to use debugging techniques:

Save the corrected script (e.g., debug_script.sh).

Make the script executable: chmod +x debug_script.sh.

Run the script: ./debug_script.sh.

Observe the output and notice the trace of the script execution with debug information.

In this example, the output should clearly show the execution steps, and we can identify any syntax errors or unexpected behavior during the script execution. After fixing the issues, you can run the script without debugging to verify the correct execution:

```
calculate_square() {
   result=$(($1 * $1))
   echo "The square of $1 is: $result"
}

# Call the function to calculate the square
calculate_square 5
```