

Assignment 1: Ensure the script checks if a specific file (e.g., myfile.txt) exists in the current directory. If it exists, print "File exists", otherwise print "File not found".

```
#!/bin/bash
```

```
if [ -f "myfile.txt" ]; then
    echo "File exists"
else
    echo "File not found"
fi
```

- `#!/bin/bash`: It tells the system which interpreter to use to execute the script.
- `[-f "myfile.txt"]; then`: The `[]` are used for conditions. `-f` is a flag that checks if the given file exists "myfile.txt" is the file we are checking for. The `;` marks the end of the condition.
- `echo`: `echo` is a command used to print text to the terminal.
- `fi`: It specifies the end of the conditional block.

Assignment 2: Write a script that reads numbers from the user until they enter '0'. The script should also print whether each number is odd or even.

```
#!/bin/bash
```

```
while true; do
    echo "Enter a number (enter '0' to exit): "
    read number
    if [ "$number" -eq 0 ]; then
        echo "Exiting..."
        break
    fi
    if [ $((number % 2)) -eq 0 ]; then
        echo "$number is even"
    else
        echo "$number is odd"
    fi
done.
```

- `#!/bin/bash`: This is the shebang line, specifying that this script should be interpreted by Bash.
- `while true; do`: The loop will continue until it encounters a `break` statement.

- `echo "Enter a number (enter '0' to exit): "`: This line prints asking the user to enter a number.
- `read number`: This command reads the user's input.
- `if ["$number" -eq 0]; then`: This checks if the entered number is '0'.
- `if [$((number % 2)) -eq 0]; then`: This checks if the number is even.
- `echo :echo` is a command used to print text to the terminal.
- `done`: This marks the end of the while loop.

Assignment 3: Create a function that takes a filename as an argument and prints the number of lines in the file. Call this function from your script with different filenames.

```
#!/bin/bash
count_lines() {
    filename="$1" # Get the filename from the first argument
    if [ -f "$filename" ]; then # Check if the file exists
        lines=$(wc -l < "$filename") # Count the lines in the file
        echo "Number of lines in $filename: $lines"
    else
        echo "File $filename not found"
    fi
}

# Call the function with different filenames
count_lines "file1.txt"
count_lines "file2.txt"
count_lines "file3.txt"
```

- `#!/bin/bash`: This is the shebang line, specifying that this script should be interpreted by Bash.
- `count_lines() { ... }`: This defines a function named `count_lines`
- `wc -l < "$filename"`: It counts the number of lines in the file using