Labwork 8 - Shell Scripting Drills

Build comfort with basic Bash control flow, arithmetic, and file management by writing a handful of short scripts. Work in a scratch directory (e.g., `~/labwork8`) unless your instructor gives other directions. Make each script executable with `chmod +x <script>.sh` and capture a brief run log showing your commands and outputs.

Conditionals & loops emphasis: Each script should clearly demonstrate `if`/`elif`/`else` branches and the requested loop types (`for` vs. `while`). Add a brief inline comment before each non-trivial conditional or loop noting its purpose (e.g., `# guard invalid range`, `# tally evens/odds`). Show in your run log where each branch/loop was exercised.

Part A - Parity and Factorials

Create `even_odd_factorial.sh` that:

1. Accepts exactly one positional argument `num`; prints `Usage: ./even_odd_factorial.sh <number>` and exits if missing.
2. Uses an `if`/`else` to print whether the input number is even or odd (show the modulus check in your code).
3. Uses a `while` loop to print factorials for every integer from 1 through `num` as `N! = value` (show the loop counter increment).
4. Uses a `for` loop to sum the even and odd integers from 1 through `num` separately and prints both subtotals (include an `if` inside the loop to split even vs. odd).
5. Demo run: `./even_odd_factorial.sh 6`.

Part B - Workspace Setup Helper

Create `labwork8_setup.sh` that automates a quick workspace prep:

1. Create `~/Labwork8/Data` if it does not exist.
2. List the contents of `~/Labwork8`, and list only its subdirectories.
3. Count how many direct subdirectories are under `~/Labwork8` and print the count.
4. `cd` into `~/Labwork8/Data` and print the working directory.
5. Create `info.txt` inside `~/Labwork8/Data` containing the provided six-line motivational text.
6. `grep` the word `key` from `info.txt` into `~/Labwork8/keywords.txt`.
7. Write the first and last lines of `info.txt` to `~/Labwork8/summary.txt`, then append the total word count of `info.txt`.
8. Rename `info.txt` to `myQuotes.txt`, and set its permissions to `755`.
9. Ensure any directory-creation logic is wrapped in an `if` that checks for existence (avoid errors when rerun).
10. Demo run: `./labwork8_setup.sh` (should succeed when rerun without errors).

Part C - Two-Number Comparison and Cubes

Create `compare_and_cubes.sh` that:

1. Accepts exactly two positional numbers; prints `Usage: ./compare_and_cubes.sh number1 number2` and exits if missing (guard with an `if` on `$#`).
2. Compares the two numbers with `if`/`elif`/`else` and prints which is larger, or states they are equal.
3. Prints cubes for 0-5 using a `for` loop, then 6-10 using a `while` loop, labeled as `Cube(N) = value`; make the loop bounds explicit in your comments.
4. Demo run: `./compare_and_cubes.sh 12 4`.

Part D - Range Squares and Parity Counts

Create `range_squares.sh` that:

1. Prompts for `start` and `end` numbers (read from stdin).
2. Uses an `if` to exit with `Invalid range` if `start` is greater than `end` (include `exit 1`).
3. Uses a `for` loop over the range to print each square and tallies how many numbers are even vs. odd (include an `if` inside the loop for parity tracking).
4. After the loop, prints `Even count: X` and `Odd count: Y`.
5. Demo run: `./range_squares.sh` then enter `3` and `9`.

Part E - Summation With a Branch

Create `sum_and_branch.sh` that:

1. Prompts for a number `num` (stdin).
2. Sums 1 through `num` using a `while` loop and prints `Sum from 1 to <num>: <sum>` (show the loop counter and accumulator updates).
3. Uses an `if`/`else` to branch: if the sum is greater than 100, divide it by 5 and print the result; otherwise, print that the sum is small with no further action.
4. Demo run: `./sum_and_branch.sh` then enter `25`.

Deliverables

- The five scripts with execute permission in your labwork directory.
- A short log or screenshots showing command lines used and sample outputs for each script (include at least the suggested demo runs).
- A note on any errors or surprises you hit while scripting or running the commands.