Objective: System Snapshot Utility via .bash_profile

You're building a utility script for your DevOps team called sysreport.sh. This script should behave like a command-line tool — the team should be able to run it with different flags to quickly check system health in a customizable way.

Your Task:

Write a Bash script (sysreport.sh) that:

1. Accepts flags to decide what to capture:
   - --cpu: Include CPU usage ✓
   - --mem: Include memory usage ✓
   - --disk: Include disk usage ✓
   - --no-log: Don’t write to a file, only display on screen ✓
2. If no flags are given, it should default to all 3 (CPU + Memory + Disk) and log enabled.
3. The log file should be named ~/sysreport_YYYYMMDD_HHMMSS.log
4. Display output on screen as well
5. Use an environment variable (SYSLOG) to store and pass the log file name.
6. Handle invalid flags with a usage message.
7. Include a timestamp at the top of the report. ✓

BREAK DOWN

1) ACCEPT FLAGS

L default case — switch case

2) Log file name — date

3) Print output on sceen — tee

4) Timestamp — date

CPU Logic — top -bn1 ; b= batch mode

Memory logic — free -m

n1 = 1 iteration

Disk logic — df -h

4) Export log file name — export

Skeleton

1) Declare Env Vars

1) CPU = false .

2) Mem= false .

3) Disk = false

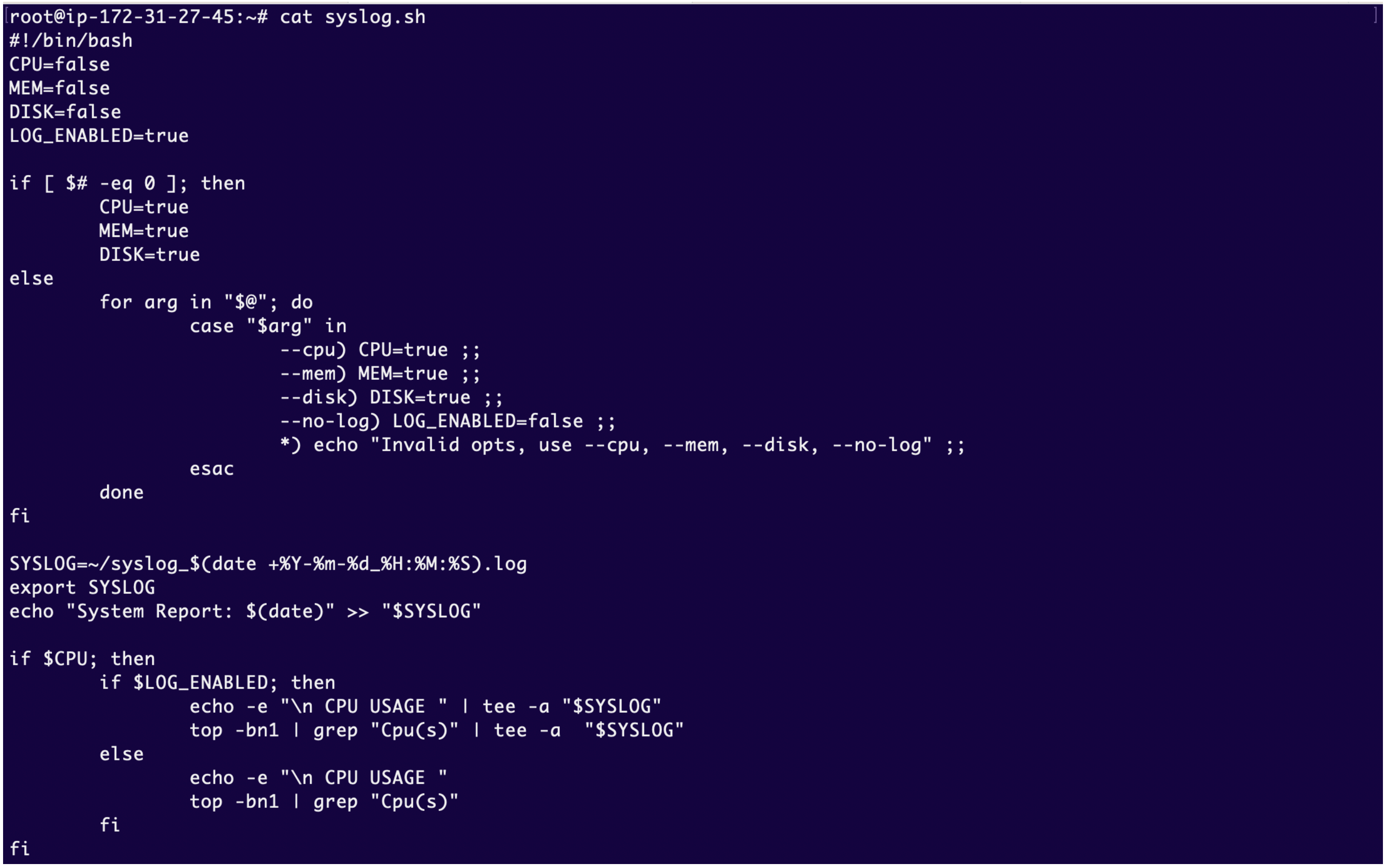
4) Log_enabled = true

2) Check if no flags are provided

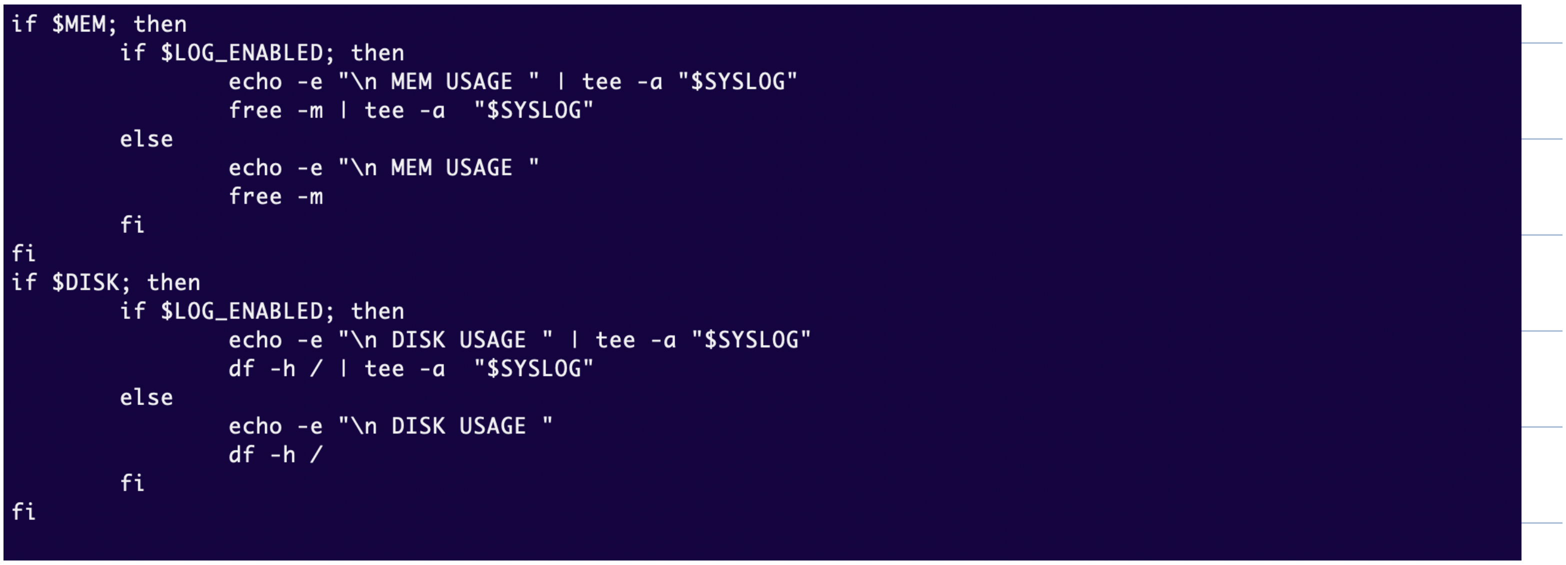
```
if [ $# -eq 0 ] then
    cpu = true
    mem = true
    disk =true
else

# Read the flags

for arg in "$@"; do
    case "$arg" in
        --cpu) cpu =true ;;
        --mem) mem = true ;;
        --disk) disk =true ;;
        --no-log) log-enabled=false ;;
        *) echo "Invalid opts, use --cpu,
            --mem, --disk, --no-log" ;;
    esac
done
```

```
fi

 # Create log file name


# Add timestamp on top of report

# Calculate the usages
        if $cpu ; then
            if log_enabled ; then
          # print cpu in terminal + syslog
            else
          # print cpu in terminal only

      if $mem



       if $disk
if  cpu = true
    if log_enabled = true
        print  in  terminal  +  logfile
    else
        print  only  in  terminal
if   mem = true
    if
    else
if  disk = true
    if
    else
```

\n → new line
\t → tab



```
root@ip-172-31-27-45:~# cat syslog.sh
#!/bin/bash
CPU=false
MEM=false
DISK=false
LOG_ENABLED=true

if [ $# -eq 0 ]; then
        CPU=true
        MEM=true
        DISK=true
else
        for arg in "$@"; do
                case "$arg" in
                        --cpu) CPU=true ;;
                        --mem) MEM=true ;;
                        --disk) DISK=true ;;
                        --no-log) LOG_ENABLED=false ;;
                        *) echo "Invalid opts, use --cpu, --mem, --disk, --no-log" ;;
                esac
        done
fi

SYSLOG=~/syslog_$(date +%Y-%m-%d_%H:%M:%S).log
export SYSLOG
echo "System Report: $(date)" >> "$SYSLOG"

if $CPU; then
        if $LOG_ENABLED; then
                echo -e "\n CPU USAGE " | tee -a "$SYSLOG"
                top -bn1 | grep "Cpu(s)" | tee -a  "$SYSLOG"
        else
                echo -e "\n CPU USAGE "
                top -bn1 | grep "Cpu(s)"
        fi
fi
```



```
if $MEM; then
        if $LOG_ENABLED; then
                echo -e "\n MEM USAGE " | tee -a "$SYSLOG"
                free -m | tee -a  "$SYSLOG"
        else
                echo -e "\n MEM USAGE "
                free -m
        fi
fi
if $DISK; then
        if $LOG_ENABLED; then
                echo -e "\n DISK USAGE " | tee -a "$SYSLOG"
                df -h / | tee -a  "$SYSLOG"
        else
                echo -e "\n DISK USAGE "
                df -h /
        fi
fi
```

fail if any variable is not set

→ set -euo pipefail

non-zero status code then exit script

exits if any pipes fails

grep 'error' | awk

FAIL FAST