**Practice Problem # 01**

1. Use `grep` to print all lines from `sample\_1.txt` that contain a magic string. A magic string is any five-character alphanumeric word composed exclusively of the letters `{e, n, r, v}` and digits `0` through `4`. The search should be case-insensitive. For each match, print the matching line number along with one line after the matching line.

| grep -inwE -A1 '[enrv0-4]{5}' sample\_directory/sample\_1.txt  # test if any matching line appears in the inverse match  grep -inwE -v '[enrv0-4]{5}' sample\_directory/sample\_1.txt |
| --- |

2.Write a single command that finds all `.log` files in the `sample\_directory` that were modified within the last 7 days, and for each file, prints the number of lines containing the word `ERROR` (case-insensitive).

| find sample\_directory -name "\*.log" -mtime -7 -exec grep -iwcH error {} \;  sample\_directory/subdir1/server.log:3  sample\_directory/app2.log:0  sample\_directory/app1.log:10 |
| --- |

3. Update your command so that it considers only the first 10 lines of each file.

- Hint 1: You can use `sh -c '<your\_command>'` within `-exec`

- Hint 2: `sh -c` can also access arguments with `$1, $2, ... `

| find sample\_directory -name "\*.log" -mtime -7 -exec \  sh -c 'echo -n "$1: "; cat $1 | head -n 10 | grep -iwc error' \_ {} \;  sample\_directory/subdir1/server.log: 3  sample\_directory/app2.log: 0  sample\_directory/app1.log: 5 |
| --- |

**Practice Problem # 02**

1. Write a shell script that takes three input integers (values are given in non-decreasing order, \*i.e.\*, $a\le b\le c$) and determines if those three values can be three sides of a right triangle (one angle is a $90^\circ$). [Hint: Use Pythagorean theorem]

| #!/bin/bash  read a  read b  read c  lhs=$((a\*a+b\*b))  rhs=$((c\*c))  if [[ $lhs -eq $rhs ]]; then  echo "Right Triangle!"  else  echo "Not a Right Triangle!"  fi |
| --- |

2. Write a bash script that takes three integer inputs and prints the largest value among them.

| # Try at home |
| --- |

3. Write a bash script that takes a year as input from the command line and determines if the given year is a leap year or not.

| #!/bin/bash  year=$1 # Take year from command line argument  if [[ $((year % 4)) -eq 0 ]]; then  if [[ $((year % 100)) -eq 0 ]]; then  if [[ $(( year % 400)) -eq 0 ]]; then  echo "Leap Year"  else  echo "Not a Leap Year"  fi  else  echo "Leap Year"  fi  else  echo "Not a Leap Year"  fi |
| --- |

**Practice Problem # 03**

1. Let's go back to the `sample\_directory`. Write a bash script that loops over all `.log` files in `sample\_directory` and its subdirectories. For each `.log` file, counts how many lines contain the word ERROR and WARNING (case-insensitive), and prints a summary line per file as below. At the end, prints the total number of ERROR lines and WARNING lines across all `.log` files.

| #!/bin/bash  total\_error=0  total\_warning=0  # Use find to get the list, then loop over it with for  for file in $(find sample\_directory -type f -name "\*.log"); do  # error\_count=$(grep -i 'ERROR' "$file" | wc -l)  # warning\_count=$(grep -i 'WARNING' "$file" | wc -l)  error\_count=$(grep -ic 'ERROR' "$file")  warning\_count=$(grep -ic 'WARNING' "$file")  echo "$file: ERROR=$error\_count, WARNING=$warning\_count"  total\_error=$((total\_error + error\_count))  total\_warning=$((total\_warning + warning\_count))  done  echo "Total ERROR lines: $total\_error"  echo "Total WARNING lines: $total\_warning" |
| --- |