Green University of Bangladesh
Department of Computer Science and Engineering(CSE)

Faculty of Sciences and Engineering

Semester: (Summer, Year:2025), B.Sc. in CSE (Day)

Lab Report 01

Course Title: Operating System Lab

Course Code: CSE 310 **Section:** 223 D2

Student Details

Name	ID
Anisur Rahaman Maruf	222902078

Submission Date: 05/07/2025

Course Teacher's Name: Umme Habiba

[For Teachers use only: Don't Write Anything inside this box]

Lab Report Status

Marks: ………………………………… **Signature:.....................**

Comments:... **Date:..............................**

1. Title

Understanding and Applying Regular Expressions in Linux

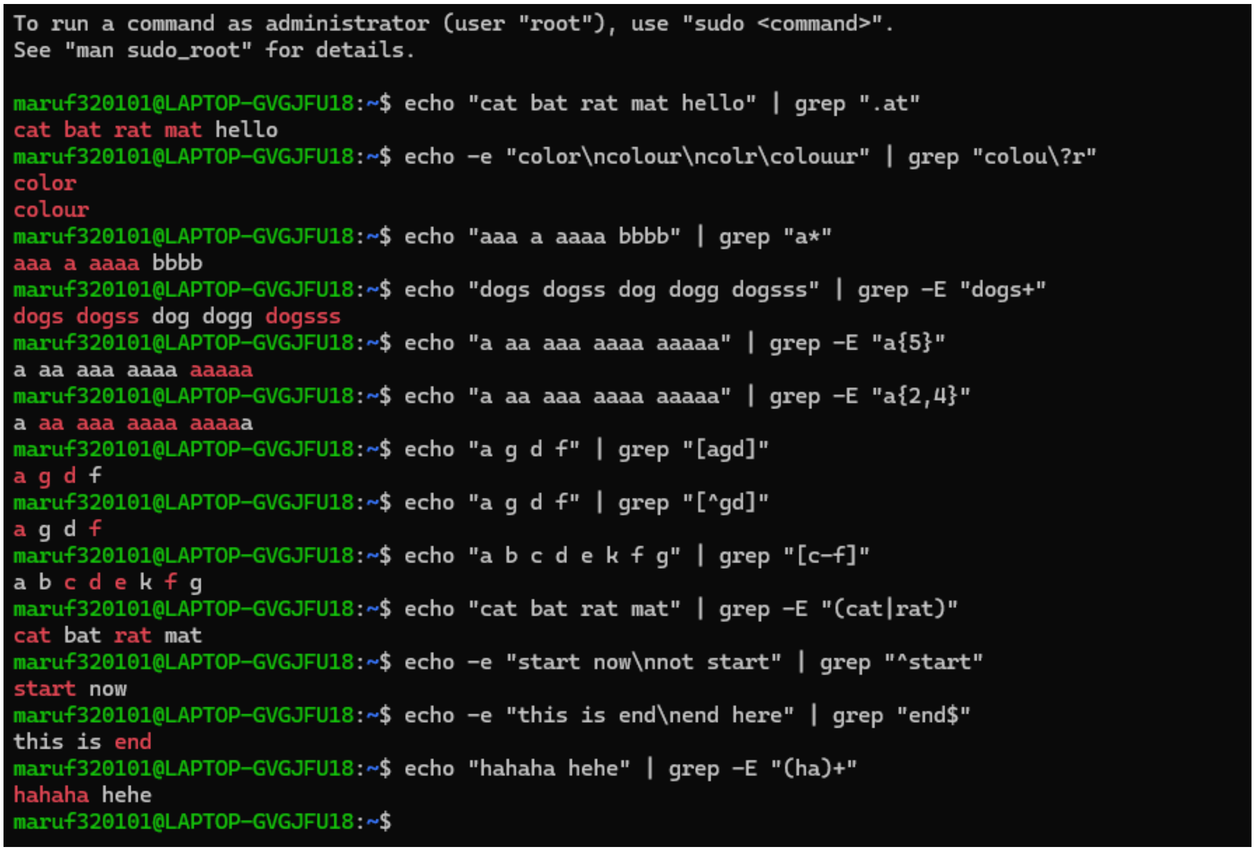
2. Introduction

Regular Expressions (commonly referred to as "regex") are powerful tools used in Linux to match specific patterns in text. They are essential for tasks like searching files (grep), stream editing (sed), and more. In this lab, we explore fundamental regex symbols and observe how they behave when applied with real Linux commands. This foundational knowledge is crucial for text processing and scripting in Linux environments.

3. Regular Expression Commands with Output

1. . — Matches any single character.
2. ? — Matches the preceding character 0 or 1 time.
3. * — Matches the preceding character 0 or more times.
4. + — Matches the preceding character 1 or more times.
5. n — Matches the preceding character exactly n times.
6. n,m — Matches the preceding character at least n times but not more than m times.
7. [agd] — Matches any one character inside the brackets (a, g, or d).
8. [^gd] — Matches any one character not inside the brackets (not g or d).
9. [c-f] — Matches any character in the range c to f (c, d, e, or f).
10. () — Groups several characters as one unit.
11. | — Acts as a logical OR between expressions.
12. ^ and $ — Matches the beginning (^) or end ($) of a line.

4. Output



```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

maruf320101@LAPTOP-GVGJFU18:~$ echo "cat bat rat mat hello" | grep ".at"
cat bat rat mat hello
maruf320101@LAPTOP-GVGJFU18:~$ echo -e "color\ncolour\ncolr\colouur" | grep "colou\?r"
color
colour
maruf320101@LAPTOP-GVGJFU18:~$ echo "aaa a aaaa bbbb" | grep "a*"
aaa a aaaa bbbb
maruf320101@LAPTOP-GVGJFU18:~$ echo "dogs dogss dog dogg dogsss" | grep -E "dogs+"
dogs dogss dog dogg dogsss
maruf320101@LAPTOP-GVGJFU18:~$ echo "a aa aaa aaaa aaaaa" | grep -E "a{5}"
a aa aaa aaaa aaaaa
maruf320101@LAPTOP-GVGJFU18:~$ echo "a aa aaa aaaa aaaaa" | grep -E "a{2,4}"
a aa aaa aaaa aaaaa
maruf320101@LAPTOP-GVGJFU18:~$ echo "a g d f" | grep "[agd]"
a g d f
maruf320101@LAPTOP-GVGJFU18:~$ echo "a g d f" | grep "[^gd]"
a g d f
maruf320101@LAPTOP-GVGJFU18:~$ echo "a b c d e k f g" | grep "[c-f]"
a b c d e k f g
maruf320101@LAPTOP-GVGJFU18:~$ echo "cat bat rat mat" | grep -E "(cat|rat)"
cat bat rat mat
maruf320101@LAPTOP-GVGJFU18:~$ echo -e "start now\nnot start" | grep "^start"
start now
maruf320101@LAPTOP-GVGJFU18:~$ echo -e "this is end\nend here" | grep "end$"
this is end
maruf320101@LAPTOP-GVGJFU18:~$ echo "hahaha hehe" | grep -E "(ha)+"
hahaha hehe
maruf320101@LAPTOP-GVGJFU18:~$
```

5. Discussion

In this lab, we explored the most frequently used regular expression patterns and their practical uses in Linux command-line tools. By combining simple regex patterns with commands like grep, users can perform complex text searches efficiently. For example, understanding the difference between *, +, and {n,m} allows fine-grained control over how many times characters must appear. Additionally, character classes and anchors (^, $) are vital for defining position-specific patterns.