

**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 NO - 01**

**Course Title: Operating System Lab**

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

**Lab Experiment Name: Regular Expression Overview**

**Student Details**

|  |  |  |
| --- | --- | --- |
| **Name** | | **ID** |
| **1.** | **MD.SHAJALAL** | **223002088** |

**Submission Date : 01 – 07 - 2025**

**Course Teacher’s Name : Umme Habiba**

|  |
| --- |
| **Lab Report Status**  **Marks: ………………………………… Signature:.....................**  **Comments:.............................................. Date:..............................** |

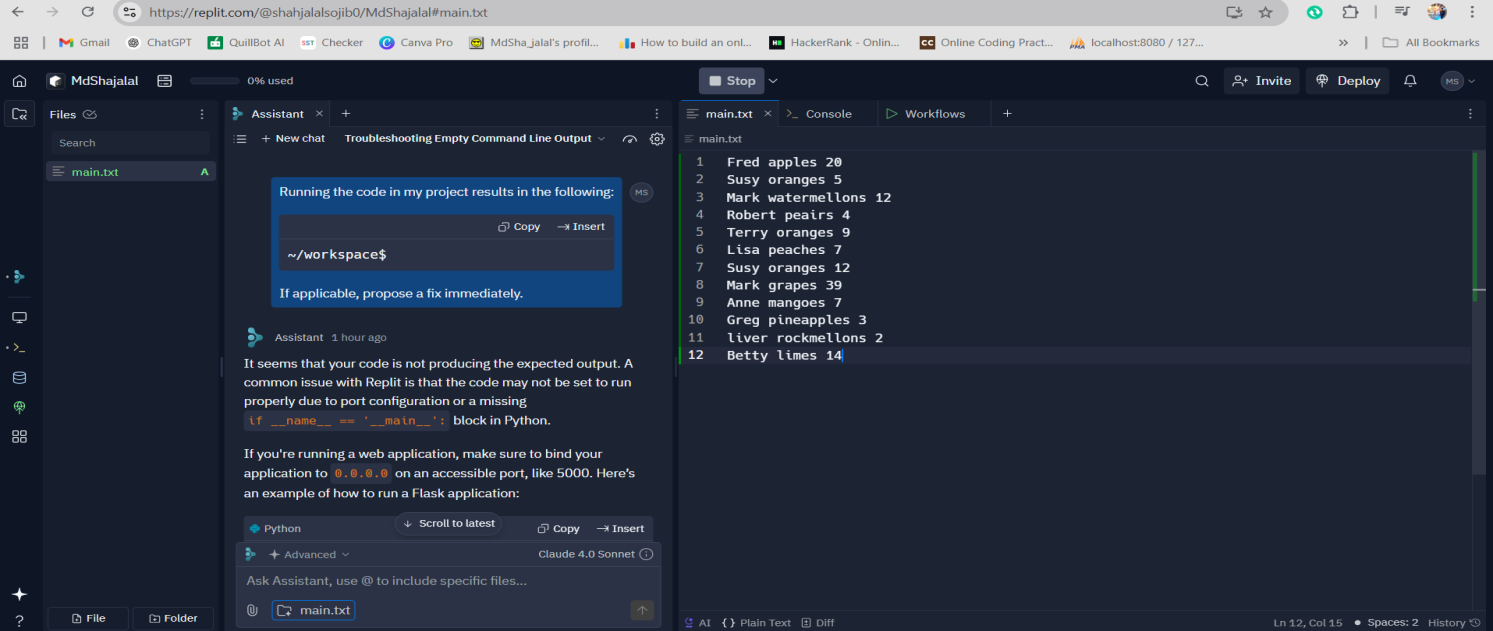
### **1.Objective:**

To understand and demonstrate the use of various **Regular Expression (Regex)** symbols in text pattern matching.

**2.Regex Symbols and Their Uses:**

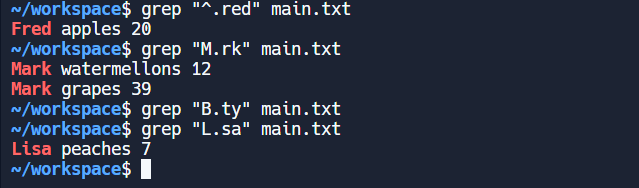
| **Symbol** | **Meaning** | **Example** | **Expected Match** |
| --- | --- | --- | --- |
| . | Matches any single character | a.c matches abc, a1c | abc, a2c |
| ? | Matches 0 or 1 of the preceding character | a?b matches b, ab | b, ab |
| \* | Matches 0 or more of the preceding character | lo\*l matches ll, lol, loool | ll, lol |
| + | Matches 1 or more of the preceding character | go+gle matches gogle, google | google |
| n | Exact number of repetitions | a{3} matches aaa | aaa |
| n,m | Range of repetitions | a{2,4} matches aa, aaa, aaaa | aa, aaa |
| [ agd] | Match any of a, g, or d | b[agd]c matches bac, bgc | bac |
| [^gd] | Exclude g and d | b[^gd]c matches bbc, not bgc | bbc |
| [c-f] | Match range from c to f | [c-f]at matches cat, dat | cat, eat |
| () | Group expressions | (ab)+ matches ab, abab | ab, abab |
| ` | ` | OR operator | `cat |
| ^ | Start of line | ^Hello matches lines starting with Hello | Hello world |
| $ | End of line | world$ matches lines ending with world | Hello world |

**3.Txt File:**

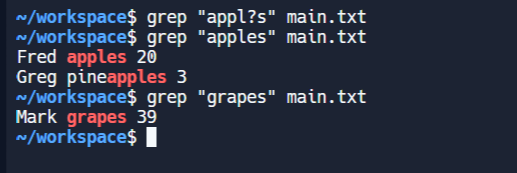
****

**4.Command Examples and Output:**

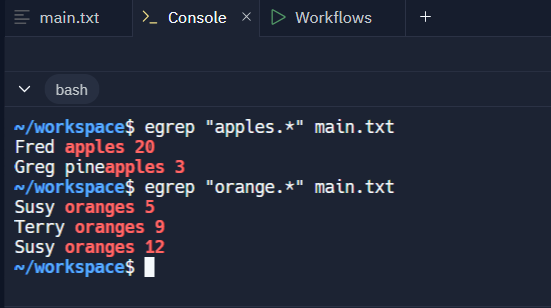
**Q1.** . (dot) - a single character.

****

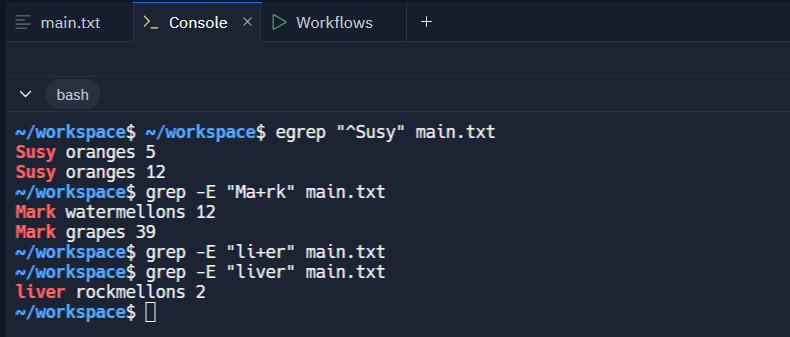
**Q2.** ? - the preceding character matches 0 or 1 times only.

****

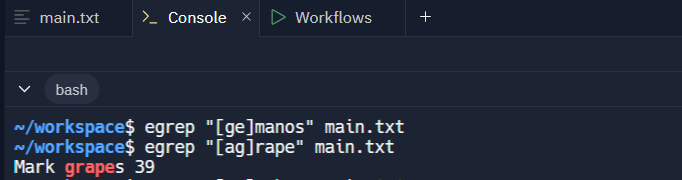
**Q3 .** \* - the preceding character matches 0 or more times.



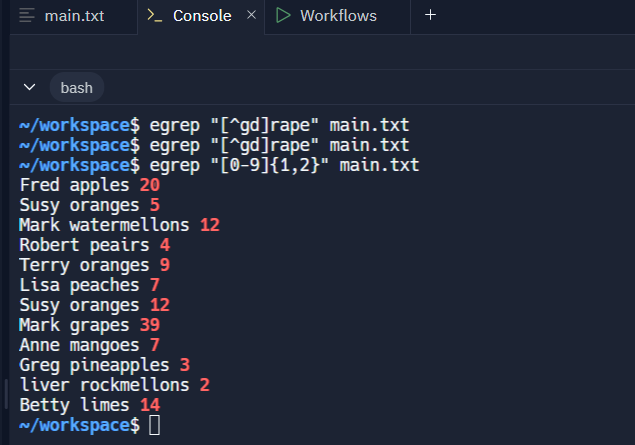
Q4. + - the preceding character matches 1 or more times.



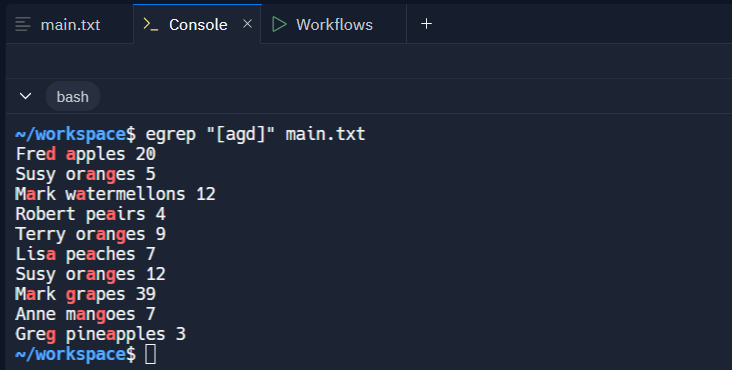
Q5. n - the preceding character matches exactly n times.



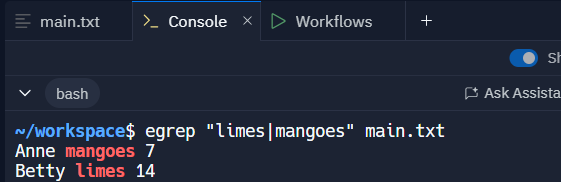
Q6. n,m - the preceding character matches at least n times and not more than m times.



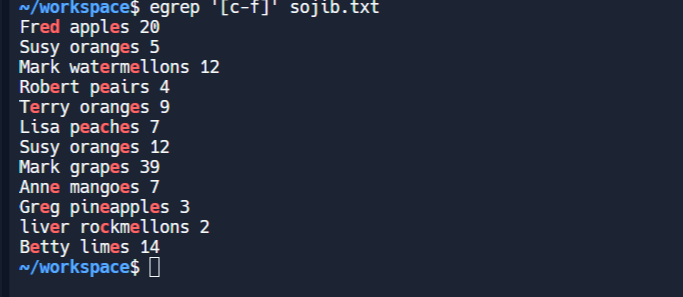
Q7. [agd] - the character is one of those included within the square brackets.



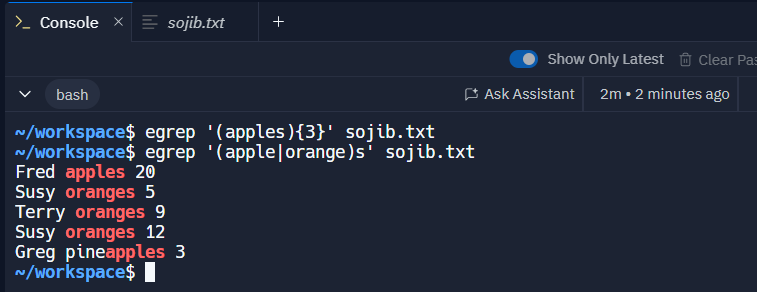
Q8. [gd] - the character is not one of those included within the square brackets.



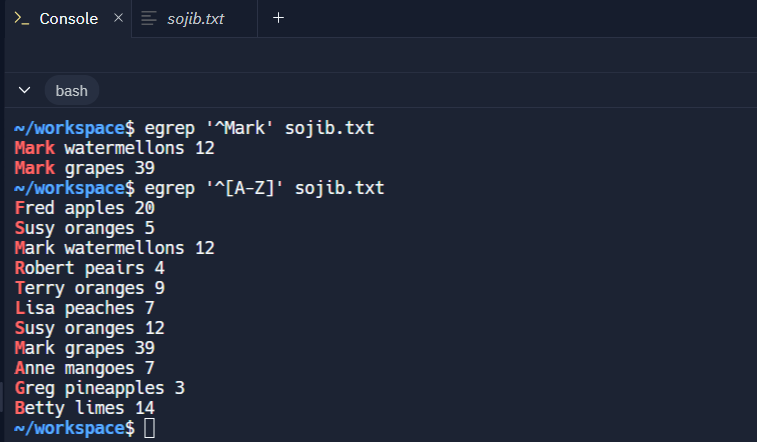
Q9. [c − f] - the dash within the square brackets operates as a range. In this case it means either the lettersc, d, e or f.



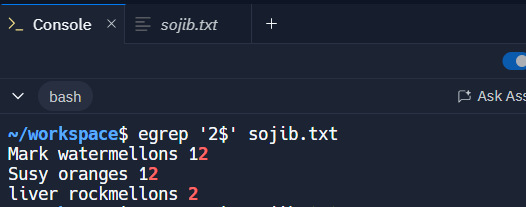
Q10. () - allows us to group several characters to behave as one.



Q11. ˆ- matches the beginning of the line.



Q12. $ - matches the end of the line.



### **Conclusion:**

In this lab, we explored the powerful pattern-matching abilities of **Regular Expressions** using the command line. Each symbol helps in identifying, searching, and processing textual data efficiently — an essential skill for system administrators and OS-level scripting.