SCS2201 : Assignment 01

Index Number : 21001512

String matching algorithm: **Rabin Karp**
Implemented language: **Python 3.11.4**

The instructions given are for Windows users, Mac and Linux might have a similar approach.

Preparing the environment to run

No need to install any additional libraries or software. Make sure you have installed a version above Python 3.11 in your PC and it is added as a path variable, so it can be accessed from the command prompt.

How to run?

- Open cmd
- Navigate to the directory where the python file is present.
- Execute "python test.py" (Use python or python3 as configured in your PC)
- The pattern and string are given as user input values

Input and output explained

- The regex patterns are limited to | , ^ , $, \b.
- The program can handle only one regex pattern in the input at a time.
- Outputs for regex patterns:
  - | : A dictionary output mentioning the string indexes where the pattern is present. An empty value signifies that there are no matching patterns.
  - ^ : Boolean value is returned based on the presence of the pattern at the beginning of the string.
  - $: Boolean value is returned based on the presence of the pattern at the beginning of the string.
  - \b : A list with the string indexes where the pattern is present. Empty list signifies that there are no matching patterns.

Explanation for choosing the string matching algorithm

- The string algorithm used here is Rabin-Karp.
- The main reason for using this algorithm is simpler implementation along with other benefits.
- Useful for large alphabets, where computing hash values is more effective than traditional string matching.
- Easy to handle and maintain the code.

Code explanation

- A hash function is defined to compute hash values
- A sliding function is defined to move on from substring to substring (python string slicing is used) and hash value of each substring is compared to the hash value of the given pattern, if the hash values match, then string matching is done.
  - When moving from one substring to another substring, the hash value changes. So the subsequent substring is sliced and the new hash value is computed, as opposed to the traditional implementation of a sliding function, where the new hash value of the new character gets added to the existing hash value and the hash value of the old character gets subtracted.
  - This approach is easier to implement and provides better code readability and maintenance.
- Conditional logic along with string slicing is used to handle the string matching based on the input regex patterns.

Test cases

- A text file with relevant test cases and the outputs is provided for reference.