Assignment: Intelligent CV Analyzer using String Matching Algorithms
Title: Design and Implementation of a CV Analyzer using String Matching Algorithms for Automated Skill Extraction and Job Fit Evaluation

Problem Statement:

Recruiters and HR departments often receive hundreds of CVs for a single job opening. Manually reviewing these CVs to identify qualified candidates is time-consuming and error prone. Most CVs contain unstructured text, making it difficult to automatically identify relevant skills, qualifications, and experience.
The challenge is to develop an **intelligent CV Analyzer** that uses string matching and pattern matching algorithms to automatically extract key information from resumes and compare it against predefined job requirements.
This project aims to explore the use of **Brute Force, Rabin–Karp, and Knuth–Morris–Pratt (KMP) algorithms** for efficient keyword and pattern matching in large collections of CV text. The system should identify whether a candidate's profile matches a given job description based on the presence (or absence) of required skills and qualifications.

Objectives:

To implement and compare classical string-matching algorithms **(Brute Force, Rabin–Karp, KMP) for text analysis.**

To develop a program that can automatically:

- Extract and match skills, tools, and technologies from CVs.
- Compare them with job description keywords.
- To analyze the efficiency and accuracy of each algorithm on large text datasets (multiple CVs).
- To recommend the most effective algorithm for real-time CV screening.

Case Study Scenario:

You are part of a recruitment software development team building a CV Screening Tool for a hiring agency. The system must process hundreds of text-based resumes daily and rank candidates based on keyword relevance to job postings.

Each job description includes a list of mandatory and optional keywords (e.g., "Python", "Machine Learning", "SQL", "Data Analysis").
The system should:

- Read a candidate's CV (plain text or extracted text from PDF or Docx).
- Use string matching algorithms to detect occurrences of these keywords.
- Compute a relevance score based on matches.
- Output a ranked list of candidates by match percentage.

Project Tasks:

Algorithm Implementation:

- *Implement Brute Force, Rabin–Karp, and KMP algorithms for pattern searching.*

Each algorithm should:

- Identify all occurrences of a keyword in the CV text.
- Support multiple pattern searches (list of keywords).
- Handle both case-sensitive and case-insensitive matches.

Dataset:

- CVs are provided (pdf and docx)
- Create 3 job descriptions with defined skill requirements (e.g., Data Scientist, Web Developer, Software Tester).

Functionality Implementation:

Input: *CV text and job description keywords.*

Output:

- List of matched and missing keywords.
- Relevance score (e.g., percentage of skills matched).
- Performance metrics (execution time, comparisons).

Performance Analysis:

Compare the efficiency of the three algorithms on:

- Small vs large CV files.
- Single keyword vs multiple keyword searches.
- Record execution time and number of character comparisons.
- Present findings using charts or tables.

Reporting:

Include:

- Algorithm explanation and pseudocode.
- Flowchart of your CV Analyzer system.
- Comparative analysis of results.
- Discussion on trade-offs (speed, accuracy, collisions, preprocessing).
- Recommendation of the best algorithm for practical use.

Expected Deliverables:

- **It must be a desktop or web application:** Source code (Python/C++/Java/.net/android, any language, tool can be used)
- CV and job description text files (datasets is already provided not to be uploaded again, job description must be uploaded)
- Output reports showing matches and scores
- Performance results (tables/graphs)

Project report (6–8 pages) covering:

- Introduction & problem definition
- System design & algorithm explanation
- Implementation details
- Experimental results & analysis
- Conclusion & future improvements