Project Title: VCF File Processing for Variant Extraction and Filtering

Aim:

To automate the parsing, extraction, and filtering of genomic variant data from a VCF (Variant Call Format) file using Python, enabling efficient downstream analysis.

Objective:

1. Extract and save all VCF header lines to a separate file.

2. Extract and save the following variant details to a new file:

- Chromosome number (CHROM)

- Position of alteration (POS)

- Reference allele (REF)

- Alternate allele (ALT)

3. Filter and save only those variant entries where the depth (DP) is exactly 6.

Methodology:

- Tools Used: Python with built-in libraries; gzip for reading compressed files.

- Data Source: A .vcf.gz file containing variant information from a genomic sample.

- Steps:

1. The script opens the compressed VCF file and reads its contents.

2. Header lines (starting with '#') are separated and saved to headers.txt.

3. Variant lines (non-header) are parsed to extract four key fields: CHROM, POS, REF, and ALT. These are saved to chr_pos_ref_alt.tsv.

4. INFO fields of each variant line are scanned to check for the presence of DP=6. Matching variants are saved to dp6_variants.tsv.

Results:

The script generates three output files:

- headers.txt - Contains all header metadata from the VCF file.

- chr_pos_ref_alt.tsv - A simplified tab-separated file listing the genomic location and allelic changes.

- dp6_variants.tsv - A filtered list of variants with depth (DP) equal to 6.

These outputs are suitable for further statistical or visual analysis of variant quality and distribution.

Conclusion:

The project successfully automated the extraction and filtering of VCF data, simplifying variant-level quality control. This type of processing is essential in genomics pipelines, especially in large-scale studies where manual inspection of VCFs is impractical.

Summary:

This script-based utility provides a foundational tool for bioinformaticians to preprocess VCF files. It isolates metadata, key variant data, and applies depth-based filtering, laying the groundwork for further annotation, visualization, or statistical evaluation in downstream workflows.