B+ Tree Project - User Guide

**1. Introduction**

This program manages a blocked sequence set and B+ Tree using CSV-based data. Its main functionalities include:

* CSV conversion into fixed-sized blocks
* Index generation for efficient lookup
* Blocked sequence file creation for structured data storage
* Dynamic record insertion, deletion, and search operations
* Debugging via dump functions to analyze physical and logical data layout

**2. Installation Instructions**

* Requirements:
  + C++ compiler (e.g., g++),
  + Linux command line environment.
  + Doxygen (for documentation generation, optional)
* Directory structure:
  + 1. Buffer.h, Buffer.cpp
    2. Record.h
    3. Block.h, BlockBuffer.h, BlockBuffer.cpp
    4. main.cpp
    5. BPlusTree.h, BPLusTree.cpp
    6. resources/zip\_codes.csv
    7. output/

**3. How to Compile and Run**

* **Compiling:**
* g++ -g ./main.cpp ./Buffer.cpp ./BlockBuffer.cpp ./BPlusTree.cpp -o btree
* **Running in Generation Mode:**

./btree

This command will:

* Read the CSV file from ./resources/zip\_codes.csv
* Convert CSV records into length-indicated format
* Group records into fixed-size blocks and write the resulting blocked sequence set file to ./output/blocked\_sequence\_set.txt
* Build the B+ Tree structure from the blocked sequence set
* **Running in Dump Mode**

./btree --dumpPhysical

* + Lists all blocks as they appear in the file.
  + Useful for verifying physical layout.
* **Logical Order (Follow Next Block Links):**

./btree --dumpLogical

* Follows nextBlock pointers for each block.
* Useful for simulating linked list traversal or logical sequence reading.
* Displays the tree in a structured format
* **Dump Both Orders Together:**

./btree --dumpPhysical --dumpLogical

**4. Troubleshooting**

* Ensure the CSV file is in the correct location (./resources/zip\_codes.csv).
* Verify that the output directory exists.
* Check file permissions if the file cannot be opened.