# Test Document — Project 2: Zip Code Lookup System

## 1. Introduction

This document describes the testing plan, test cases, expected results, and verification methods for the Zip Code Lookup System developed in the Project-2-Zip-Code-Team-4 repository. The goal is to verify that the program correctly reads, indexes, and retrieves ZIP code records from the length-indicated data file using the primary key index file.

## 2. Test Objectives

• Validate correct index loading and record retrieval from the data file.

• Confirm that only required components (header, index, one record) are loaded into memory.

• Ensure consistent functionality regardless of CSV column order.

• Test robustness against invalid or missing data.

• Verify correct output formatting and error messages.

## 3. Test Environment

|  |  |
| --- | --- |
| Component | Description |
| System | Linux / macOS / Windows (CLI supported) |
| Language | C++17 or higher |
| Compiler | g++ / clang++ |
| Dependencies | None (standard libraries only) |
| Input Files | us\_postal\_codes.csv, randomized.csv, zipdata.lenind, zipdata.index |
| Executable | project\_part2.exe |

## 4. Test Scenarios

### Test Case 1: Valid ZIP Code Lookup

Objective: Verify correct retrieval of existing ZIP code record.

Input: ./project\_part2.exe -Z56301

Expected Output:  
ZIP: 56301  
City: St. Cloud  
State: MN  
...

Pass Criteria: Record is found, fields correctly labeled and match source data.

### Test Case 2: Nonexistent ZIP Code

Objective: Ensure proper handling when ZIP code not found in index.

Input: ./project\_part2.exe -Z99999

Expected Output:  
ZIP code 99999 not found in the data file.

Pass Criteria: No crash, proper error message displayed.

### Test Case 3: Randomized CSV File

Objective: Confirm output consistency when CSV columns are randomized.

Input: Generate and compare outputs from normal and randomized data files.

Expected Output:  
Identical outputs across datasets.

Pass Criteria: Record outputs are consistent across CSV versions.

### Test Case 4: Column Reordering Robustness

Objective: Verify program works with reordered CSV columns.

Input: Reorder columns manually and regenerate data file.

Expected Output:  
Output remains identical.

Pass Criteria: Results unaffected by column order.

### Test Case 5: Missing or Corrupted Data File

Objective: Ensure graceful failure when data file is missing.

Input: ./project\_part2.exe -Z56301 (data file missing)

Expected Output:  
Error: Unable to open data file zipdata.lenind.

Pass Criteria: Program exits safely with clear error message.

### Test Case 6: Missing or Corrupted Index File

Objective: Ensure graceful failure when index file is missing or corrupted.

Input: ./project\_part2.exe -Z56301 (index file missing)

Expected Output:  
Error: Unable to load primary key index file.

Pass Criteria: Program exits safely with clear error message.

### Test Case 7: Edge ZIP Codes

Objective: Test boundary values for first and last ZIP codes.

Input: ./project\_part2.exe -Z00501  
./project\_part2.exe -Z99950

Expected Output:  
Valid results if records exist.

Pass Criteria: No buffer overflow, handles boundaries correctly.

### Test Case 8: Performance Check

Objective: Measure efficiency with large data files.

Input: ./project\_part2.exe -Z10001

Expected Output:  
Response time under 0.5 seconds.

Pass Criteria: Program efficient and responsive.

## 5. Test Data Summary

|  |  |
| --- | --- |
| File | Description |
| us\_postal\_codes\_ROWS\_RANDOMIZED\_length\_indicated\_header\_record.txt | Randomized version for column robustness tests |

## 6. Expected Results Summary

|  |  |  |  |
| --- | --- | --- | --- |
| Test ID | Description | Expected Result | Pass/Fail |
| TC-01 | Valid ZIP | Record displayed correctly | PASS |
| TC-02 | Nonexistent ZIP | Not found message | PASS |
| TC-03 | Randomized CSV | Same output as normal | PASS |
| TC-04 | Column reorder | Correct results | PASS |
| TC-05 | Missing data file | Error message | PASS |
| TC-06 | Missing index | Error message | PASS |
| TC-07 | Edge ZIP | Boundary handled | PASS |
| TC-08 | Performance | Fast response | PASS |

## 7. Conclusion

Testing verifies that the system correctly reads from a structured binary data file, uses index-based lookups for efficiency, handles invalid cases gracefully, and maintains robustness to CSV variations. The application passes functional, boundary, and performance tests.