//Design Considerations

There should be two separate “programs” (main()), one to read the text data file and write a reformatted file to be read by the second program which will create the report to standard out. The format of the reformatted file can be any form such as text, binary, csv, Serialized, JSON, html, or other form of your choosing. Note before the report is displayed, a single line with “File: ” then the path of the input file for the report is displayed.

The first program will have 3 run-time parameters passed into the program via the command line, the data source file path, the destination file path, and the number of records in the data file, SmallAreaIncomePovertyEstData.

The second program will have 2 run-time parameters, the input file path and the number of records.

//Summary Report Code

```
import java.io.*;

public class SummaryReport {
    public static void main(String[] args) {
        if (args.length != 3) {
            System.out.println("Usage: SummaryReport C:\\Users\\rdcox\\Documents\\JAVA\\SmallAreaIncomePoverty\\SmallAreaIncomePovertyEstData.dat C:\\Users\\rdcox\\Documents\\JAVA\\SmallAreaIncomePoverty\\SummaryReport.csv 13487");
            return;
        }

               String inputFilePath = args[0];
        String outputFilePath = args[1];
        int numRecords = Integer.parseInt(args[2]); // Number of records from file

        System.out.println("Input File: " + inputFilePath);
        System.out.println("Output File: " + outputFilePath);
        System.out.println("Number of Records: " + numRecords);

        try (BufferedReader reader = new BufferedReader(new FileReader(inputFilePath));
             BufferedWriter writer = new BufferedWriter(new FileWriter(outputFilePath))) {

            String line;
            int recordCount = 0;

            // Write CSV header to the output file
            writer.write("State, District ID, District Name, Population, Child Population, Child Poverty Population, Percentage of Children in Poverty\n");

            while ((line = reader.readLine()) != null && recordCount < numRecords) {
                String state = line.substring(0, 2).trim();
                String districtID = line.substring(3, 8).trim();
                String districtName = line.substring(9, 80).trim();

                // Extract string data for population, child population, and child poverty population
                String populationStr = line.substring(82, 90).trim();
                String childPopulationStr = line.substring(91, 99).trim();
                String childPovertyPopulationStr = line.substring(100, 108).trim();
                // Variables to store parsed integer values
                int population = 0;
                int childPopulation = 0;
                int childPovertyPopulation = 0;

                // Validate and convert to integers, or set default values if invalid
                if (!populationStr.isEmpty() && populationStr.matches("\\d+")) {
                    population = Integer.parseInt(populationStr);
                }

                if (!childPopulationStr.isEmpty() && childPopulationStr.matches("\\d+")) {
                    childPopulation = Integer.parseInt(childPopulationStr);
                }

                if (!childPovertyPopulationStr.isEmpty() && childPovertyPopulationStr.matches("\\d+")) {
                    childPovertyPopulation = Integer.parseInt(childPovertyPopulationStr);
                }

                        float percentageChildrenInPoverty = (childPopulation > 0) ? ((float) childPovertyPopulation / childPopulation) * 100 : 0.0f;

                        // Format the percentage to two decimal places as a String
                        String formattedPercentage = String.format("%.2f", percentageChildrenInPoverty);

                        // Process or output the extracted data (for demonstration)
                        System.out.println("State: " + state);

                        System.out.println("District ID: " + districtID);

                        System.out.println("District Name: " + districtName);

                        System.out.println("Population: " + population);

                        System.out.println("Child Population: " + childPopulation);

                        System.out.println("Child Poverty Population: " + childPovertyPopulation);

                        System.out.println("Percentage of Children in Poverty: " + formattedPercentage + "%");


                        // Write the extracted data to the output file

                        String formattedData = String.format("%s, %s, %s, %d, %d, %d, %s%%\n", state,
        districtID, districtName, population, childPopulation, childPovertyPopulation, formattedPercentage);

                        writer.write(formattedData);





                recordCount++;

              }

            } catch (IOException | NumberFormatException e) {

              e.printStackTrace();

            }

          }

        }



        //Data Formatter Code

        import java.io.*;
```

```
public class DataFormatter {
    public static void main(String[] args) {
        if (args.length != 2) {
            System.out.println("Usage: DataFormatter
C:\\Users\\rdcox\\Documents\\JAVA\\SmallAreaIncomePoverty\\SmallAreaIncomePovertyEstData.dat
13487");
            return;
        }


         String inputFilePath = args[0];
        int numRecords = Integer.parseInt(args[1]); // Number of records


        try (BufferedReader reader = new BufferedReader(new FileReader(inputFilePath))) {
            int recordCount = 0;
            while (recordCount < numRecords && reader.ready()) {
                String line = reader.readLine();


                // Check the length of the line before extracting substrings
                if (line.length() >= 108) { // Ensure the line is long enough for the expected substring extraction
                    String state = line.substring(0, 2).trim();
                    String districtID = line.substring(3, 8).trim();
                    String districtName = line.substring(9, 80).trim();
                    String population = line.substring(82, 90).trim();
                    String childPopulation = line.substring(91, 99).trim();
                    String childPovertyPopulation = line.substring(100, 108).trim();
```

```java
                    // Format data into CSV-like columns and output to the console
                    String formattedData = String.format("%s, %s, %s, %s, %s, %s\n", state, districtID, districtName, population, childPopulation, childPovertyPopulation);
                    System.out.print(formattedData);
                }

                recordCount++;
            }
        } catch (IOException | NumberFormatException e) {
            e.printStackTrace();
        }
    }
}
```