JAVA LAB **Date : 21.02.2025**

FILE HANDLING

1. Write student names and scores to a file named temp.txt and reads the data back from the file.

2. Write a java program to write the user inputted name and its corresponding phone number in each line of the file separated by a blank space. Write a java program to search for the given phone number in the given file and print the name.

3. Suppose you want to back up a huge file (e.g., a 10-GB AVI file) to a CD-R. You can achieve it by splitting the file into smaller pieces and backing up these pieces separately. Write a utility program that splits a large file into smaller ones using the following command:

   java Exercise SourceFile numberOfPieces

   The command creates the files **SourceFile.1**, **SourceFile.2**, . . . ,**SourceFile.n**, where **n** is **numberOfPieces** and the output files are about the same size.

4. Encode the file by adding **10** to every byte in the file. Write a program that prompts the user to enter an input file name and an output file name and saves the encrypted version of the input file to the output file.

5. Implement a class named **BitOutputStream**, as shown in Figure, for writing bits to an output stream. The **writeBit (char bit)** method stores the bit in a byte variable. When you create a **BitOutputStream**, the byte is empty. After invoking **writeBit('1')**, the byte becomes **00000001**. After invoking **writeBit("0101")**, the byte becomes **00010101**. The first three bits are not filled yet. When a byte is full, it is sent to the output stream. Now the byte is reset to empty. You must close the stream by invoking the **close()** method. If the byte is neither empty nor full, the **close()** method first fills the zeros to make a full **8** bits in the byte and then outputs the byte and closes the stream. Write a test program that sends the bits **0100001001000001001101** to the file named **Exercise1.dat**.