Shri Ramdeobaba College of Engineering & Management Nagpur-13

Department of Computer Application

Session: 2023-2024

Submission for

Course Name: Design Analysis and Algorithm Lab

Course Code: MCP546

Name of the Student: Jayesh Lalit Nandanwar

Class Roll No: 26

Semester: MCA II semester

Shift: 2

Batch: 2

Under the Guidance of

Prof. Manda Ukey

Date of submission: 09/03/2024

Practical 5

Aim: Perform MERGE sort on the data sets that you have created in practical_0.

Display the time taken to sort the elements from the files in ascending order. Consider random repeated and random unrepeated files.

Compare its time with the time taken for selection, insertion sort.

For file 1 (Sequential unrepeated numbers):

Code:
```
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.Scanner;

public class MergeSortFile1 {
    public static void main(String[] args) throws IOException {
        String file_name="sequentialUnrepeatedNumbers";
        FileReader f = new FileReader("./"+file_name+".txt");

        // Reading numbers from file
        Scanner fileScanner = new Scanner(f);
        int[] array = new int[100001];
        int size = 0;

        while (fileScanner.hasNextInt()) {
            array[size++] = fileScanner.nextInt();
        }
        fileScanner.close();

        long start = System.currentTimeMillis();
        mergeSort(array);
        long finish = System.currentTimeMillis();
        long timeElapsed = finish - start;

        writeToFile(array,file_name);

        System.out.println("\nTime taken for sorting: " + timeElapsed + "
milliseconds");
    }

    public static void mergeSort(int[] array) {
        if (array == null) {
            return;
```

```
}

if (array.length > 1) {
    int mid = array.length / 2;

    // Split left part
    int[] left = new int[mid];
    for (int i = 0; i < mid; i++) {
        left[i] = array[i];
    }

    // Split right part
    int[] right = new int[array.length - mid];
    for (int i = mid; i < array.length; i++) {
        right[i - mid] = array[i];
    }
    mergeSort(left);
    mergeSort(right);

    int i = 0;
    int j = 0;
    int k = 0;

    // Merge left and right arrays
    while (i < left.length && j < right.length) {
        if (left[i] < right[j]) {
            array[k] = left[i];
            i++;
        } else {
            array[k] = right[j];
            j++;
        }
        k++;
    }
    // Collect remaining elements
    while (i < left.length) {
        array[k] = left[i];
        i++;
        k++;
    }
    while (j < right.length) {
        array[k] = right[j];
        j++;
        k++;
    }
}
```

```
    }

    public static void writeToFile(int arr[],String file_name) throws IOException {
        FileWriter writer = new FileWriter("./"+file_name+"SortedOutput.txt");
        for (int i = 0; i < arr.length - 1; i++) {
            writer.write(arr[i] + "\n");
        }
        writer.close();
    }
}
```

Output:

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms
/PSWindows

PS D:\Work_Files\RCOEM\DAA_Lab\Practical_5>  & 'C:\Program Files\Java\jre-1.8\b
in\java.exe' '-cp' 'C:\Users\MSI\AppData\Roaming\Code\User\workspaceStorage\ec5
8122c06276737688ba8fbaf31e2f9\redhat.java\jdt_ws\Practical_5_da3308f8\bin' 'Mer
geSortFile1'

Time taken for sorting: 24 milliseconds
PS D:\Work_Files\RCOEM\DAA_Lab\Practical_5>
```

Time Taken: 24 ms

For file 2 (Random unrepeated numbers):

Code:
```
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.Scanner;

public class MergeSortFile2 {
   public static void main(String[] args) throws IOException {
      String file_name="randomUnrepeatedNumbers";
      FileReader f = new FileReader("./"+file_name+".txt");

      // Reading numbers from file
      Scanner fileScanner = new Scanner(f);
      int[] array = new int[100001];
      int size = 0;

      while (fileScanner.hasNextInt()) {
         array[size++] = fileScanner.nextInt();
      }
      fileScanner.close();

      long start = System.currentTimeMillis();
      mergeSort(array);
      long finish = System.currentTimeMillis();
      long timeElapsed = finish - start;

      writeToFile(array,file_name);

      System.out.println("\nTime taken for sorting: " + timeElapsed + "
milliseconds");
   }

   public static void mergeSort(int[] array) {
      if (array == null) {
         return;
      }

      if (array.length > 1) {
         int mid = array.length / 2;

         // Split left part
         int[] left = new int[mid];
         for (int i = 0; i < mid; i++) {
            left[i] = array[i];
         }
```

```
        // Split right part
        int[] right = new int[array.length - mid];
        for (int i = mid; i < array.length; i++) {
            right[i - mid] = array[i];
        }
        mergeSort(left);
        mergeSort(right);

        int i = 0;
        int j = 0;
        int k = 0;

        // Merge left and right arrays
        while (i < left.length && j < right.length) {
            if (left[i] < right[j]) {
                array[k] = left[i];
                i++;
            } else {
                array[k] = right[j];
                j++;
            }
            k++;
        }
        // Collect remaining elements
        while (i < left.length) {
            array[k] = left[i];
            i++;
            k++;
        }
        while (j < right.length) {
            array[k] = right[j];
            j++;
            k++;
        }
    }
}

public static void writeToFile(int arr[],String file_name) throws IOException {
    FileWriter writer = new FileWriter("./"+file_name+"SortedOutput.txt");
    for (int i = 0; i < arr.length - 1; i++) {
        writer.write(arr[i] + "\n");
    }
    writer.close();
}
}
```

Output:

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms
/PSWindows

PS D:\Work_Files\RCOEM\DAA_Lab\Practical_5>  & 'C:\Program Files\Java\jre-1.8\b
in\java.exe' '-cp' 'C:\Users\MSI\AppData\Roaming\Code\User\workspaceStorage\ec5
8122c06276737688ba8fbaf31e2f9\redhat.java\jdt_ws\Practical_5_da3308f8\bin' 'Mer
geSortFile2'

Time taken for sorting: 28 milliseconds
PS D:\Work_Files\RCOEM\DAA_Lab\Practical_5>
```

Time Taken: 28 ms

For file 3 (Random repeated numbers):

Code:
```
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.Scanner;

public class MergeSortFile3 {
   public static void main(String[] args) throws IOException {
      String file_name="randomRepeatedNumbers";
      FileReader f = new FileReader("./"+file_name+".txt");

      // Reading numbers from file
      Scanner fileScanner = new Scanner(f);
      int[] array = new int[100001];
      int size = 0;

      while (fileScanner.hasNextInt()) {
         array[size++] = fileScanner.nextInt();
      }
      fileScanner.close();

      long start = System.currentTimeMillis();
      mergeSort(array);
      long finish = System.currentTimeMillis();
      long timeElapsed = finish - start;

      writeToFile(array,file_name);

      System.out.println("\nTime taken for sorting: " + timeElapsed + "
milliseconds");
   }

   public static void mergeSort(int[] array) {
      if (array == null) {
         return;
      }

      if (array.length > 1) {
         int mid = array.length / 2;

         // Split left part
         int[] left = new int[mid];
         for (int i = 0; i < mid; i++) {
            left[i] = array[i];
         }
```

```
        // Split right part
        int[] right = new int[array.length - mid];
        for (int i = mid; i < array.length; i++) {
            right[i - mid] = array[i];
        }
        mergeSort(left);
        mergeSort(right);

        int i = 0;
        int j = 0;
        int k = 0;

        // Merge left and right arrays
        while (i < left.length && j < right.length) {
            if (left[i] < right[j]) {
                array[k] = left[i];
                i++;
            } else {
                array[k] = right[j];
                j++;
            }
            k++;
        }
        // Collect remaining elements
        while (i < left.length) {
            array[k] = left[i];
            i++;
            k++;
        }
        while (j < right.length) {
            array[k] = right[j];
            j++;
            k++;
        }
    }
}

public static void writeToFile(int arr[],String file_name) throws IOException {
    FileWriter writer = new FileWriter("./"+file_name+"SortedOutput.txt");
    for (int i = 0; i < arr.length - 1; i++) {
        writer.write(arr[i] + "\n");
    }
    writer.close();
}
}
```

Output:

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms
/PSWindows

PS D:\Work_Files\RCOEM\DAA_Lab\Practical_5>  & 'C:\Program Files\Java\jre-1.8\b
in\java.exe' '-cp' 'C:\Users\MSI\AppData\Roaming\Code\User\workspaceStorage\ec5
8122c06276737688ba8fbaf31e2f9\redhat.java\jdt_ws\Practical_5_da3308f8\bin' 'Mer
geSortFile3'

Time taken for sorting: 27 milliseconds
PS D:\Work_Files\RCOEM\DAA_Lab\Practical_5>
```

Time Taken: 27 ms

Comparison between time taken:

Sorting Algorithm	**Time Taken (in ms) For Sorting**		
	Sequential Unrepeated Numbers	Random Unrepeated Numbers	Random Repeated Numbers
Selection Sort	1103	1591	1690
Insertion Sort	3	1830	1900
Merge Sort	24	28	27

Observations:

- Time taken to sort Random Unrepeated Numbers and Random Repeated Numbers is minimum in using Merge Sort Algorithm.
- Time taken to sort Sequential Unrepeated Numbers is less in Insertion Sort (3 ms) than Selection Sort (1103 ms) and Merge Sort(24 ms)