

**CSC248**

**FUNDAMENTALS OF DATA  
STRUCTURE**

**LAB ASSIGNMENT 1**

NAME : MUHAMMAD REDZA BIN MAHAYADIN

STUDENT ID : 2022676696

GROUP : RCDCS1103B

LECTURER : SIR MOHD NIZAM BIN OSMAN

Land Class

public class Land {

    private String id;

    private String ownerName;

    private char type;

    public double area;

    public Land(){

        this.id = "";

        this.ownerName = "";

        this.type = ' ';

        this.area = 0;

    }

    public Land(String id, String ownerName, char type, double area){

        this.id = id;

        this.ownerName = ownerName;

        this.type = type;

        this.area = area;

    }

    public String getId(){

        return this.id;

    }

    public String getOwnerName(){

        return this.ownerName;

    }

    public char getType(){

        return this.type;

    }

    public double getArea(){

        return this.area;

    }

    public void setId(String id){

        this.id = id;

    }

    public void setOwnerName(String ownerName){

        this.ownerName = ownerName;

    }

    public void setType(char type){

        this.type = type;

    }

    public void setArea(double area){

        this.area = area;

    }

    public String toString(){

        return ("ID  : " + id + "\nName: " + ownerName + "\nType: " + type + "\nArea: " + area);

    }

    public double calcTax(){

        double taxRate = 0;

        switch (type) {

            case 't':

            case 'T':

                taxRate = 10;

                break;

            case 's':

            case 'S':

                taxRate = 15;

                break;

            case 'b':

            case 'B':

                taxRate = 20;

                break;

            case 'c':

            case 'C':

                taxRate = 30;

                break;

            default:

                System.out.println("Invalid type");

                break;

        }

        return (area \* taxRate);

    }

}

App Class

import java.util.Scanner;

import java.io.\*;

public class App {

    public static void main(String[] args) {

        Scanner in = new Scanner(System.in);

        Land[] lands = null;

        try {

            BufferedReader br = new BufferedReader(new FileReader("customerData.txt"));

            // Count the number of lines in the file

            int count = 0;

            String line = br.readLine();

            while (line != null) {

                count++;

                line = br.readLine();

            }

            br.close();

            lands = new Land[count];

            // Read data into lands array

            br = new BufferedReader(new FileReader("customerData.txt"));

            int i = 0;

            String inData;

            while ((inData = br.readLine()) != null) {

                String[] tokens = inData.split(";");

                String id = tokens[0];

                String ownerName = tokens[1];

                char type = tokens[2].charAt(0);

                double area = Double.parseDouble(tokens[3]);

                lands[i] = new Land(id, ownerName, type, area);

                i++;

            } // end while

        } catch (IOException e) {

            System.out.println("Error: " + e.getMessage());

        }

        // Menu selection

        int option = 0;

        while (option != 4) {

            System.out.println("\n\t Menu Selection");

            System.out.println("1. Sort Tax Price");

            System.out.println("2. Sort Owner ID");

            System.out.println("3. Search Owner ID");

            System.out.println("4. Exit");

            System.out.print("\nYour Option: ");

            option = in.nextInt();

            System.out.println("\n----------------------------------");

            switch (option) {

                case 1:

                    System.out.println("\tSorting tax price\n");

                    bubbleSort(lands);

                    System.out.println("\tSorted using Bubble Sort");

                    System.out.println("----------------------------------");

                    break;

                case 2:

                    System.out.println("\tSorting owner ID\n");

                    insertionSort(lands);

                    for (Land land : lands) {

                        System.out.println(land.toString());

                        System.out.printf("Tax : RM%,.2f%n", land.calcTax());

                        System.out.println();

                    }

                    System.out.println("\tSorted using Insertion Sort");

                    System.out.println("----------------------------------");

                    break;

                case 3:

                    System.out.println("\tSearching owner ID\n");

                    System.out.print("Enter id: ");

                    String id = in.next();

                    System.out.println();

                    binarySearch(lands, id);

                    System.out.println("\tSearched using Binary Search");

                    System.out.println("----------------------------------");

                    break;

                case 4:

                    System.out.println("\n\tThank you for using this program.\n");

                    break;

                default:

                    System.out.println("Invalid option. Please try again.");

                    break;

            }

        }

        in.close();

    } // end main

    public static void bubbleSort(Land[] lands) {

        for (int i = 0; i < lands.length; i++) {

            for (int j = 0; j < lands.length - 1 - i; j++) {

                if (lands[j].calcTax() > lands[j + 1].calcTax()) {

                    Land temp = lands[j];

                    lands[j] = lands[j + 1];

                    lands[j + 1] = temp;

                }

            }

        }

        // Print sorted lands and their taxes

        for (Land land : lands) {

            System.out.println(land.toString());

            System.out.printf("Tax : RM%,.2f%n", land.calcTax());

            System.out.println();

        }

    } // end bubbleSort

    public static Land[] insertionSort(Land[] lands) {

        for (int i = 1; i < lands.length; i++) {

            Land temp = lands[i];

            int j = i - 1;

            while (j >= 0 && lands[j].getId().compareTo(temp.getId()) > 0) {

                lands[j + 1] = lands[j];

                j--;

            }

            lands[j + 1] = temp;

        }

        return lands;

    } // end insertionSort

    public static void binarySearch(Land[] lands, String id) {

        insertionSort(lands);

        int low = 0;

        int high = lands.length - 1;

        int mid = (low + high) / 2;

        while (low <= high) {

            if (lands[mid].getId().compareTo(id) < 0) {

                low = mid + 1;

            } else if (lands[mid].getId().compareTo(id) == 0) {

                System.out.println(lands[mid].toString());

                System.out.printf("Tax : RM%,.2f%n", lands[mid].calcTax());

                System.out.println();

                return;

            } else {

                high = mid - 1;

            }

            mid = (low + high) / 2;

        }

        System.out.println();

        System.out.println("ID not found.\n");

    } // end binarySearch

} // end class