**CSC248 – Fundamentals of Data Structure**

**Academic Session Oct 2023 – Feb 2024**

**Lab Assignment – Review of OOP**

|  |  |  |  |
| --- | --- | --- | --- |
| **Course Outcomes (CO)** | **LO1** | **LO2** | **LO3** |
| CO1 |  |  |  |
| CO2 | √ | √ | √ |
| CO3 |  |  |  |

1.1 Class Land has the following attributes and methods:

Attributes:

* id
* owner name
* house type
* area

1. Write the Land class and the following methods:

* 1. Default constructor.
  2. Normal constructor that set all data with values given through the parameter.
  3. Mutator/Setter method
  4. Retriever method for each attribute.
  5. Printer method using toString()defined method.
  6. A processor method to calculate and return the tax amount. The tax of this type of land depends on its area, and the type of the house built on the land as shown in the following table:

|  |  |  |
| --- | --- | --- |
| **House Type** | **Description** | **Tax rate (RM/m3)** |
| T | Terrace | 10 |
| S | Semi-Detached | 15 |
| B | Bungalow | 20 |
| C | Condominium | 30 |

Details of land

public class Land {

    private String id;

    private String ownerName;

    private char houseType;

    private double area;

    public Land() {

        id = "";

        ownerName = "";

        houseType = ' ';

        area = 0.0;

    }

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

        this.id = id;

        this.ownerName = ownerName;

        this.houseType = houseType;

        this.area = area;

    }

    public String getId() {

        return this.id;

    }

    public void setId(String id) {

        this.id = id;

    }

    public String getOwnerName() {

        return this.ownerName;

    }

    public void setOwnerName(String ownerName) {

        this.ownerName = ownerName;

    }

    public char getHouseType() {

        return this.houseType;

    }

    public void setHouseType(char houseType) {

        this.houseType = houseType;

    }

    public double getArea() {

        return this.area;

    }

    public void setArea(double area) {

        this.area = area;

    }

    public String toString() {

        return "ID: " + id + "\nOwner Name: " + ownerName + "\nHouse Type: " + houseType + "\nArea: " + area;

    }

    public double calculateTax() {

        double tax = 0.0;

        if (houseType == 'T') {

            tax = 10 \* area;

        } else if (houseType == 'S') {

            tax = 15 \* area;

        } else if (houseType == 'B') {

            tax = 20 \* area;

        } else if (houseType == 'C') {

            tax = 30 \* area;

        }

        return tax;

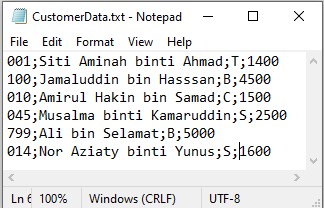
    }

}

1. Write an application program that performs the following:

* 1. Declare array of object for Land objects.

* 1. Given the input file named customerData.txt that includes the customers data such as id, owner name, house type and area. The following input text fil includes all record of customer for the Land class:



Write a program that reads each record from customerData.txt and store onto array of object Land.

* 1. Display a menu selection to select the following process:

------------------------------------------------------

Menu Selection

1. Sorting using Bubble Sort 2. Sorting using Insertion Sort 3. Searching using Binary Search

Your Option: XX

# ------------------------------------------------------

**\*\*\*\*Details explanation:**

1-Sorting using Bubble Sort – Sort the list based on the tax price and display the list

2.Sorting using Insertion Sort – Sort the list based on id and display the list

3.Searching using Binary Search-Search the item from the list based on id and display the information detail.

import java.io.BufferedReader;

import java.io.FileReader;

import java.util.Scanner;

public class Main {

    public static void main(String[] args) {

        Scanner strInput = new Scanner(System.in);

        Scanner intInput = new Scanner(System.in);

        Land[] lands = null;

        // read customerData.txt

        try {

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

            // get 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];

            count = 0;

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

            line = br.readLine();

            while (line != null) {

                String id = line.split(";")[0];

                String ownerName = line.split(";")[1];

                char houseType = line.split(";")[2].charAt(0);

                double area = Double.parseDouble(line.split(";")[3]);

                lands[count] = new Land(id, ownerName, houseType, area);

                count++;

                line = br.readLine();

            }

            br.close();

        } catch (Exception e) {

            System.out.println(e.getMessage() + "\nAt line: " + e.getStackTrace()[0].getLineNumber());

        }

        while (true) {

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

            System.out.println("Menu Selection");

            System.out.println("\n1. Sorting using Bubble Sort");

            System.out.println("2. Sorting using Insertion Sort");

            System.out.println("3. Searching using Binary Search");

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

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

            int option = intInput.nextInt();

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

            if (option == 1) {

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

                bubbleSort(lands);

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

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

                    System.out.printf("Tax: RM %,.2f\n", lands[i].calculateTax());

                    System.out.println();

                }

                System.out.println("This is sorted based on the tax price");

            } else if (option == 2) {

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

                insertionSort(lands);

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

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

                    System.out.printf("Tax: RM %,.2f\n", lands[i].calculateTax());

                    System.out.println();

                }

                System.out.println("This is sorted based on the ID");

            } else if (option == 3) {

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

                System.out.println("Enter the ID to search: ");

                String id = strInput.nextLine();

                Land land = binarySearch(lands, id);

                if (land != null) {

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

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

                } else {

                    System.out.println("Land not found!");

                }

            } else if (option == 4) {

                System.out.println("Thank you for using this program!");

                break;

            }

            while (option < 1 || option > 4) {

                System.out.println("Invalid option!");

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

                option = intInput.nextInt();

            }

            System.out.println("Enter any key to continue...");

            strInput.nextLine();

        }

        strInput.close();

        intInput.close();

    }

    // bubble sort method

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

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

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

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

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

                    Land temp = lands[j];

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

                    lands[j + 1] = temp;

                }

            }

        }

        return lands;

    }

    // insertion sort method

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

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

            Land key = lands[i];

            int j = i - 1;

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

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

                j--;

            }

            lands[j + 1] = key;

        }

        return lands;

    }

    // binary search method

    public static Land 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) {

                return lands[mid];

            } else {

                high = mid - 1;

            }

            mid = (low + high) / 2;

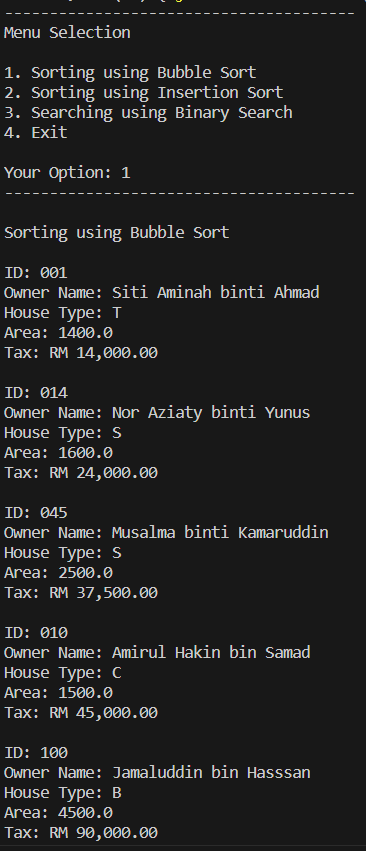
        }

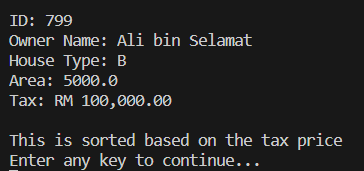
        return null;

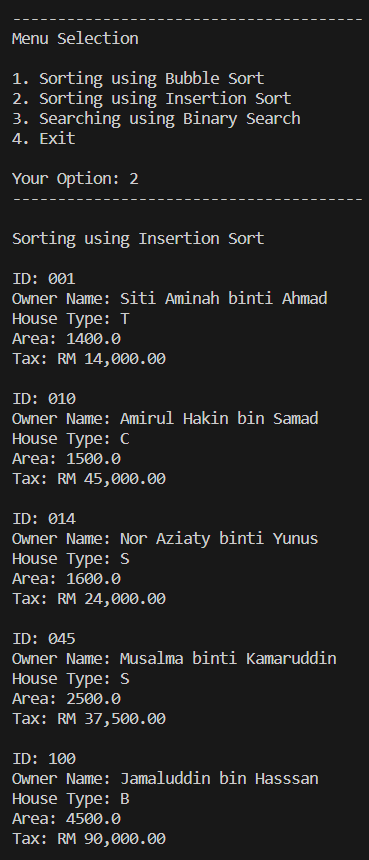
    }

}

**Sample Input/Output**

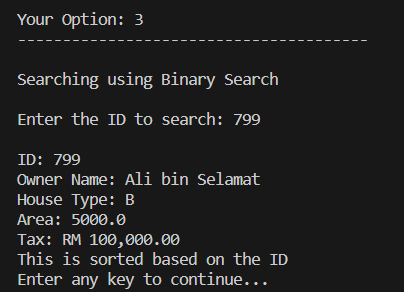
****

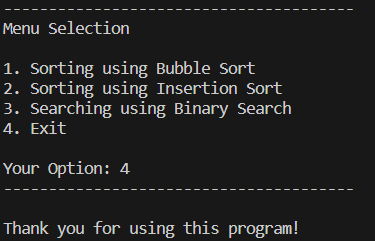
****

****

**A screenshot of a computer

Description automatically generated**

****

****