

**CSC248**

**FUNDAMENTALS OF DATA  
STRUCTURE**

**LAB ASSIGNMENT 3**

NAME : MUHAMMAD REDZA BIN MAHAYADIN

STUDENT ID : 2022676696

GROUP : RCDCS1103B

LECTURER : SIR MOHD NIZAM BIN OSMAN

FriendApp Class

import java.util.Scanner;

import java.util.LinkedList;

public class FriendApp {

    public static int choice;

    static Scanner in = new Scanner(System.in);

    static Scanner in1 = new Scanner(System.in);

    public static void main(String[] args) {

        LinkedList<Friend> sList = new LinkedList<Friend>();

        menuSelection();

        do {

            switch (choice) {

                case 1:

                    System.out.println();

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

                    int id = in1.nextInt();

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

                    String name = in.nextLine();

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

                    String hp = in.nextLine();

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

                    String email = in.nextLine();

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

                    int ind = in1.nextInt();

                    Friend f = new Friend(id, name, hp, email);

                    // Verify no duplicates

                    for (Friend friend : sList) {

                        if (friend.getId() == id) {

                            System.out.println("\nFriend ID already exists.");

                            break;

                        }

                    }

                    try {

                        sList.add(ind, f);

                        System.out.println("\nFriend added.");

                    } catch (IndexOutOfBoundsException e) {

                        sList.add(f);

                        ind = sList.indexOf(f);

                        System.out.println(

                                "\nDue to index out of bounds, the new name was added at the end of the list instead, which is at index "

                                        + ind);

                    }

                    break;

                case 2:

                    boolean found = false;

                    if (sList.size() == 0) {

                        System.out.println("\nList is empty.");

                    } else {

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

                        int id1 = in1.nextInt();

                        for (Friend friend : sList) {

                            if (friend.getId() == id1) {

                                found = true;

                                System.out.println();

                                System.out.println("Index: " + sList.indexOf(friend));

                                System.out.println(friend);

                            }

                        }

                        if (!found) {

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

                        }

                    }

                    break;

                case 3:

                    boolean found2 = false;

                    if (sList.size() == 0) {

                        System.out.println("\nAction not allowed. List is empty.");

                    } else if (sList.size() == 1) {

                        System.out.println("\nAction not allowed. List has only 1 friend.");

                    } else {

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

                        int id2 = in1.nextInt();

                        for (Friend friend : sList) {

                            if (friend.getId() == id2) {

                                found2 = true;

                                System.out.println("Are you sure you want to remove this friend? (Y/N)");

                                String confirm = in.nextLine();

                                if (confirm.equalsIgnoreCase("Y")) {

                                    sList.remove(friend);

                                    System.out.println("\nFriend removed.");

                                } else {

                                    System.out.println("\nFriend not removed.");

                                }

                            }

                        }

                        if (!found2) {

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

                        }

                    }

                    break;

                case 4:

                    boolean found3 = false;

                    if (sList.size() == 0) {

                        System.out.println("\nAction not allowed. List is empty.");

                    } else {

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

                        int id3 = in1.nextInt();

                        for (Friend friend : sList) {

                            if (friend.getId() == id3) {

                                found3 = true;

                                System.out.print("Enter new HP   : ");

                                String hp1 = in.nextLine();

                                System.out.print("Enter new email: ");

                                String email1 = in.nextLine();

                                friend.setHP(hp1);

                                friend.setEmail(email1);

                                System.out.println("\nFriend updated.");

                                break;

                            }

                        }

                        if (!found3) {

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

                        }

                    }

                    break;

                case 5:

                    if (sList.size() == 0) {

                        System.out.println("\nList is empty.");

                    } else {

                        System.out.print("\n\tList of friends");

                        for (Friend friend : sList) {

                            System.out.println();

                            System.out.println("Index: " + sList.indexOf(friend));

                            System.out.println(friend);

                        }

                    }

                    break;

                default:

                    System.out.println("\nInvalid choice.");

                    break;

            }

            System.out.println();

            menuSelection();

        } while (choice != 6);

    }

    public static void menuSelection() {

        System.out.println(

                "\n\tMenu Selection\n1. Add Friend\n2. View Friend\n3. Remove Friend\n4. Update Friend\n5. Display All Friends\n6. Exit");

        System.out.print("\nEnter choice: ");

        choice = in1.nextInt();

    }

}

Friend Class

public class Friend {

    private int idno;

    private String name, hpno, email;

    public Friend(int id, String n, String hp, String em) {

        this.idno = id;

        this.name = n;

        this.hpno = hp;

        this.email = em;

    }

    public int getId() {

        return this.idno;

    }

    public String getName() {

        return this.name;

    }

    public String getHP() {

        return this.hpno;

    }

    public String getEmail() {

        return this.email;

    }

    public void setName(String n) {

        this.name = n;

    }

    public void setHP(String hp) {

        this.hpno = hp;

    }

    public void setEmail(String mail) {

        this.email = mail;

    }

    public String toString() {

        return "ID   : " + this.idno + "\nName : " + this.name + "\nHP   : " + this.hpno + "\nEmail: " + this.email;

    }

}