/\* QUESTION 1:

Write a program to perform string operations using ArrayList. Write functions for the

following

Append - add at end

Insert – add at particular index

Find the index of a particular element (Search)

Display the list

List all string starts with given letter

List of all string contains the Substring

Sort the elements in ArrayList

Remove a particular element

Replace one string with another string in ArrayList

Remove duplicate elements

\*/

import java.util.\*;

import java.util.Collections;

import java.util.LinkedHashSet;

public class ArrayListString

{

public void append(String str,ArrayList<String> al)

{

al.add(str);

}

public void appendWithIndex(String str,int indx,ArrayList<String> al)

{

al.add(indx,str);

}

public void display(ArrayList<String> al)

{

System.out.println("Elements of the List : "+al);

}

public int findIndex(String str,ArrayList<String> al)

{

return al.indexOf(str);

}

public void startWithLtr(char ch,ArrayList<String> al)

{

Iterator<String> itr = al.iterator();

while(itr.hasNext())

{

String ele=itr.next();

char chr=ele.charAt(0);

if(ch==chr)

System.out.println(itr.next());

}

}

public void sort(ArrayList<String> al)

{

Collections.sort(al);

System.out.println("Sorted List: "+al);

}

public int removeString(String str,ArrayList<String> al)

{

Iterator<String> itr = al.iterator();

while(itr.hasNext())

{

String element = itr.next();

if(element.equals(str))

{

while(al.indexOf(str)!=-1)

al.remove(al.indexOf(str));

System.out.println("Element Removed!");

return 1;

}

}

return -1;

}

public int replaceString(String str1,String str2,ArrayList<String> al)

{

int indx=al.indexOf(str1);

if(indx!=-1)

{

al.set(indx,str2);

return 1;

}

else

return -1;

}

public void removeDuplicates(ArrayList<String> al){

LinkedHashSet<String> hashSet = new LinkedHashSet<>(al);

ArrayList<String> listWithoutDuplicates = new ArrayList<>(hashSet);

System.out.println("List after Removing Duplicates :

"+listWithoutDuplicates);

}

public void findSubstring(String str,ArrayList<String> al){

int flag=0;

Iterator<String> itr = al.iterator();

System.out.println("Elements : ");

while(itr.hasNext()) {

String element = itr.next();

if(element.contains(str)){

System.out.print(" "+element);

flag=1;

}

}

if(flag==0)

System.out.println("Substring not found");

}

public static void main(String args[]){

int wish,opt,indx,itr=0;

boolean found;

String str,ch;

char chr;

Scanner obj=new Scanner(System.in);

ArrayListString als=new ArrayListString();

ArrayList<String> al=new ArrayList<String>();

System.out.println("\n\tString Operations using ArrayList\n");

do{

System.out.println("\nEnter the Option: \n\t1 to Append - Add at End\

n\t2 to Insert – Add at Particular Index\n\t3 to Find the Index of a Particular Element\n\

t4 to Display the List\n\t5 to List all Strings starting with given Letter\n\t6 to List all

String Containing a Substring\n\t7 to Sort the Elements in ArrayList\n\t8 to Remove a

Particular Element\n\t9 to Replace one string with another string in ArrayList\n\t10 to

Remove the Duplicate Elements: ");

opt=obj.nextInt();

switch(opt){

case 1 :

System.out.println("\nEnter String to Add: ");

obj.nextLine();

str=obj.nextLine();itr++;

als.append(str,al);

break;

case 2:

System.out.println("Enter Index to Insert (From 0 to

"+itr+"): ");

indx=obj.nextInt();

System.out.println("\nEnter String to Add : ");

obj.nextLine();

str=obj.nextLine();

als.appendWithIndex(str,indx,al);

break;

case 3:

System.out.println("\nEnter String to Find: ");

obj.nextLine();

str=obj.nextLine();

indx=als.findIndex(str,al);

if(indx==-1)

System.out.println("Given String is Not Present in

the ArrayList.");

else

System.out.println("Index: "+indx);

break;

case 4:

als.display(al);

break;

case 5:

System.out.println("Enter a Character: ");

chr=obj.next().charAt(0);

als.startWithLtr(chr,al);

break;

case 6:

System.out.println("Enter the Sub-String to Find: ");

obj.nextLine();

ch=obj.nextLine();

als.findSubstring(ch,al);

break;

case 7:

als.sort(al);

break;

case 8:

System.out.println("\nEnter String to Find: ");

obj.nextLine();

str=obj.nextLine();

indx=als.removeString(str,al);

if(indx==-1)

System.out.println("Given String is not present in

the ArrayList");

break;

case 9:

System.out.println("\nEnter String to Find: ");

obj.nextLine();

str=obj.nextLine();

System.out.println("\nEnter String to Replace with: ");

ch=obj.nextLine();

indx=als.replaceString(str,ch,al);

if(indx==-1)

System.out.println("Given String is not present in

the ArrayList");

else

System.out.println("Element Replaced

successfully!");

break;

case 10:

als.removeDuplicates(al);

break;

default:

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

}

System.out.println("\nDo you wish to continue?(1-Yes/2-No): ");

wish=obj.nextInt();

}while(wish!=2);

}

}

/\* QUESTION 2:

Write a program to get two integer ArrayList. Perform the following operations

Merge the two lists

Find Union of two lists

Find Intersection of two lists

Compare two lists

\*/

import java.util.\*;

import java.util.Collections;

import java.util.HashSet;

public class ArrayListInteger{

public static void main(String args[]){

int opt,wish,num;

boolean equal;

System.out.println("\n\tInteger Manipulation using ArrayList \n");

Scanner obj=new Scanner(System.in);

ArrayListInteger ali=new ArrayListInteger();

ArrayList<Integer> al1=new ArrayList<Integer>();

ArrayList<Integer> al2=new ArrayList<Integer>();

do{

System.out.println("Enter An Option :\n\t1.Enter Elements to Lists\n\

t2.Merge the 2 Lists\n\t3.Find Union of the two Lists\n\t4.Find Intersection of the two

Lists\n\t5.Compare the two Lists\n\tEnter Your Choice: ");

opt=obj.nextInt();

switch(opt){

case 1:

System.out.println("\nEnter Integer to Add : ");

System.out.println("\nList 1 : ");

num=obj.nextInt();

al1.add(num);

System.out.println("\nList 2 : ");

num=obj.nextInt();

al2.add(num);

break;

case 2:

al1.addAll(al2);

System.out.println("List After Merging : "+al1);

break;

case 3:

Set<Integer> set = new HashSet<Integer>();

set.addAll(al1);

set.addAll(al2);

System.out.println("Union of the two Lists : "+set);

break;

case 4:

List<Integer> list = new ArrayList<Integer>();

for(Integer t : al1){

if(al2.contains(t))

list.add(t);

}

System.out.println("Intersection of the two Lists : "+list);

break;

case 5:

equal=al1.equals(al2);

if(equal==true)

System.out.println("The two Lists are Equal");

else

System.out.println("The two Lists are Not Equal");

break;

default:

System.out.println("\n\tInvalid Input.");

}

System.out.println("\nDo You wish to continue ?(1-Yes/2-No): ");

wish=obj.nextInt();

}while(wish!=2);

}

}

/\* QUESTION 3:

Using Collection framework, create a doubly linked list of integers and perform the

following operations.

a. Insert element on both sides

b. Delete element on both sides

c. Insert an element at a particular position

d. Delete a particular element

e. Search for a particular element

f.Display list in forward order and backward order

g.Sort the elements in LinkedList

h. Replace one element in the list with another list

i.Remove duplicate elements

\*/

import java.util.\*;

public class DLL{

public static void main(String args[]){

LinkedList<Integer> list = new LinkedList<>();

Iterator it;

Iterator looper;

Scanner s=new Scanner(System.in);

int x,y,i,count;

int opt=1;

while(opt!=0){

System.out.println("\n\t\tLINKED LIST MANIPULATION\n");

System.out.println("Enter An Option\n\t1 to Insert At Front\n\t2 to

Insert At End\n\t3 to Insert Before Element\n\t4 to Delete An Element\n\t5 to Search An

Element\n\t6 to Display the List in Forward\n\t7 to Display the List in Reverse\n\t8 to

Sort the List\n\t9 to Replace An Element with Another\n\t10 to Remove Duplicates\n\t0

to Exit\n\tYour Choice: ");

opt=s.nextInt();

switch(opt){

case 1:

System.out.println("Enter an Element: ");

x=s.nextInt();

list.addFirst(x);

break;

case 2:

System.out.println("Enter an Element: ");

x=s.nextInt();

list.add(x);

break;

case 3:

System.out.println("Enter an Element: ");

x=s.nextInt();

System.out.println("Enter Element Before Which to

Insert "+x);

y=s.nextInt();

i=list.indexOf(y);

if(i==-1)

System.out.println(y+" was not found in the

Linked List.\n Element "+x+" was not inserted.");

else

list.add(i,x);

break;

case 4:

System.out.println("Enter an Element: ");

x=s.nextInt();

while(list.indexOf(x)!=-1){

y=list.indexOf(x);

list.remove(y);

}

System.out.println("Element "+x+" was succesfully

removed from the Linked List.");

break;

case 5:

System.out.println("Enter an Element: ");

x=s.nextInt();

i=list.indexOf(x);

if(i==-1)

System.out.println("Element "+x+" was not found

in the Linked List.");

else

System.out.println("Element "+x+" was found at

Index "+i);

break;

case 6:

it = list.listIterator();

System.out.println("Elements in Forward Order: ");

while (it.hasNext())

System.out.print(it.next()+ " ");

break;

case 7:

it = list.descendingIterator();

System.out.println("Elements in Backward Order: ");

while (it.hasNext())

System.out.print(it.next()+ " ");

break;

case 8:

Collections.sort(list);

System.out.println("Sorted List: \n");

it = list.listIterator();

System.out.println("Elements in Forward Order: ");

while (it.hasNext())

System.out.print(it.next()+ " ");

break;

case 9:

System.out.println("Enter Replacing Element: ");

x=s.nextInt();

System.out.println("Enter Element to be replaced by

"+x);

y=s.nextInt();

while(list.indexOf(y)!=-1){

list.set(list.indexOf(y),x);

}

System.out.println("All instances of "+y+" have been

replaced by "+x+".");

break;

case 10:

LinkedList<Integer> temp = new LinkedList<>();

int size=list.size();

for(i=0;i<size;i++){

x=list.get(i);

if(!temp.contains(x))

temp.add(x);

}

list=temp;

System.out.println("Duplicates have been removed!");

break;

case 0:

System.out.println("\t\tThank You!");

break;

default:

System.out.println("\t\tInvalid Option!");

break;

}

}

}

}