



PMR: This program was easy to complete because it was the same as the past two programs. I enjoyed creating this program. I have enjoyed this module and the past module because they have been fun and easy to complete.

/\*\*

\* The Contact interface

\*

\* @author Anika Jallipalli

\*

\*/

public class Contact

{

//instance variables

private String name;

private String relation;

private String bday;

private String phone;

private String email;

//constructor for contact objects

public Contact(String n, String r, String b, String p, String e)

{

//initialize the instance variables with input data

name = n;

relation = r;

bday = b;

phone = p;

email = e;

}

public String getName()

{

return name;

}

public String getRelation()

{

return relation;

}

public String getBDay()

{

return bday;

}

public String getBMonth()

{

String b = bday.substring(0,3);

return b;

}

public String getPhone()

{

return phone;

}

public String getEmail()

{

return email;

}

public String toString()

{

String a = getName() + " " + getRelation() + " " + getBDay() + " " + getPhone() + " " + getEmail();

return a;

}

}

/\*\*

\* Run the contact list program

\*

\* @author Anika Jallipalli

\*

\*/

public class TestContact

{

public static void main(String[] args)

{

int test = 0;

Contact[] contacts = new Contact[6];

contacts[0] = new Contact("John Carter","brother","Mar 3","(342)555-7069","jcarter@carter.com");

contacts[1] = new Contact("Elise Carter","mom","Apr 19","(342)555-7011","carterMom@carter.com");

contacts[2] = new Contact("Ellie Carter","me","Jun 10","(342)555-8102","ecarter@carter.com");

contacts[3] = new Contact("Sue Ellen","friend","Mar 9","(341)555-9182","susieE@hotmail.com");

contacts[4] = new Contact("Frank Carter","dad","Dec 1","(342)555-7011","carterDad@carter.com");

contacts[5] = new Contact("Johnnie","friend","Jan 21","(341)555-7789","jDawg5555@yahoo.com");

System.out.println(" Contact List");

System.out.println();

printContacts(contacts);

//search names

System.out.println("Find Name - Johnnie");

sortNames(contacts);

test = findByName(contacts, "Johnnie");

if (test != -1)

{

System.out.println("Found:" + contacts[test].toString());

}

else

{

System.out.println("Not found.");

}

System.out.println();

System.out.println("Find Name - Sam Parker");

sortNames(contacts);

test = findByName(contacts, "Sam Parker");

if (test != -1)

{

System.out.println("Found:" + contacts[test].toString());

}

else

{

System.out.println("Not found.");

}

//search relations

System.out.println();

findByRelation(contacts,"friend");

findByRelation(contacts,"Aunt");

//search phones

findByPhone(contacts,"(333)555-8989");

findByPhone(contacts,"(342)555-7011");

//search birthdays

findByBMonth(contacts,"May");

findByBMonth(contacts,"Mar");

//search emails

System.out.println("Find Email - rgoodman@hotmail.com");

contacts = sortEmail(contacts);

test = findByEmail(contacts, "rgoodman@hotmail.com");

if (test != -1)

{

System.out.println("Found:" + contacts[test].toString());

}

else

{

System.out.println("Not found.");

}

System.out.println();System.out.println("Find Email - susieE@hotmail.com");

contacts = sortEmail(contacts);

test = findByEmail(contacts, "susieE@hotmail.com");

if (test != -1)

{

System.out.println("Found:" + contacts[test].toString());

}

else

{

System.out.println("Not found.");

}

}

//prints the arraylist

public static void printContacts(Contact[] c)

{

System.out.println("Name Relation Birthday Phone Email");

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

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

{

System.out.println(c[i]);

}

System.out.println();

}

//insertion sorts the arraylist by names

public static Contact[] sortNames(Contact[] c)

{

Contact[] newlist = new Contact[c.length];

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

{

String next = c[i].getName();

int insert = 0;

int k =i;

while(k>0 && insert == 0)

{

if(next.compareTo( newlist[k-1].getName() ) > 0)

{

insert = k;

}

else

{

newlist[k] = newlist[k-1];

}

k--;

}

newlist[insert]=c[i];

}

return newlist;

}

//binary search for a name

public static int findByName(Contact[] r, String toFind )

{

int high = r.length;

int low = -1;

int probe;

while ( high - low > 1 )

{

probe = ( high + low ) / 2;

if ( r[probe].getName().compareTo(toFind) > 0)

{

high = probe;

}

else

{

low = probe;

}

}

if ( (low >= 0) && (r[low].getName().compareTo(toFind) == 0 ))

{

return low;

}

else

{

return -1;

}

}

//find people by their relation

public static void findByRelation(Contact[] r, String toFind)

{

int found = 0;

System.out.println("Find Relation - " + toFind);

System.out.println("Find Results:");

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

{

if (r[i].getRelation().compareTo(toFind) == 0)

{

System.out.println("Found: " + r[i].toString());

found++;

}

}

if (found == 0)

{

System.out.println("There are no listings for " + toFind);

}

else

{

System.out.println("There were " + found + " listings for " + toFind);

}

System.out.println();

}

//find people by what month they were born in

public static void findByBMonth(Contact[] r, String toFind)

{

int found = 0;

System.out.println("Find Bday - " + toFind);

System.out.println("Find Results:");

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

{

if (r[i].getBMonth().compareTo(toFind) == 0)

{

System.out.println("Found: " + r[i].toString());

found++;

}

}

if (found == 0)

{

System.out.println("There are no listings for " + toFind);

}

else

{

System.out.println("There were " + found + " listings for " + toFind);

}

System.out.println();

}

//find people by their phone number

public static void findByPhone(Contact[] r, String toFind)

{

int found = 0;

System.out.println("Find Phone - " + toFind);

System.out.println("Find Results:");

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

{

if (r[i].getPhone().compareTo(toFind) == 0)

{

System.out.println("Found: " + r[i].toString());

found++;

}

}

if (found == 0)

{

System.out.println("There are no listings for " + toFind);

}

else

{

System.out.println("There were " + found + " listings for " + toFind);

}

System.out.println();

}

//insertion sorts the arraylist by email

public static Contact[] sortEmail(Contact[] c)

{

Contact[] newlist = new Contact[c.length];

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

{

String next = c[i].getEmail();

int insert = 0;

int k =i;

while(k>0 && insert == 0)

{

if(next.compareTo( newlist[k-1].getEmail() ) > 0)

{

insert = k;

}

else

{

newlist[k] = newlist[k-1];

}

k--;

}

newlist[insert]=c[i];

}

return newlist;

}

//searches emails

public static int findByEmail(Contact[] r, String toFind )

{

int high = r.length;

int low = -1;

int probe;

while ( high - low > 1 )

{

probe = ( high + low ) / 2;

if ( r[probe].getEmail().compareTo(toFind) > 0)

{

high = probe;

}

else

{

low = probe;

}

}

if ( (low >= 0) && (r[low].getEmail().compareTo(toFind) == 0 ))

{

return low;

}

else

{

return -1;

}

}

}