/\* Grader program for CS310 SP 2015 Program #2

Alan Riggins

\*/

import data\_structures.\*;

import java.util.Iterator;

public class P2Grader {

private LinearListADT<Person> list;

private LinearListADT<Integer> list2;

private String [] fnames = {"John", "Robert", "Sam", "Henry", "Bill", "William",

"Alan","Cyril","Gregory","Dennis","David","James","Joseph","Jerome","Gary"};

private String [] lnames = {"Doe","Jones","Roberts","Shapiro","Martinez","Perez",

"Wong","Nguyen","White","Bell","Brown","Black","Green","Trenton","Nagy"};

private Person [] fullName;

private Person sentinal = new Person("Aaron Burr");

public P2Grader() {

fullName = new Person[100];

for(int i=0; i < 100; i++) {

int ind1 = (int) (15.0f\*Math.random());

int ind2 = (int) (15.0f\*Math.random());

fullName[i] = new Person(fnames[ind1] + " " + lnames[ind2]);

}

list = new LinearList<Person>();

runTests();

}

private void runTests() {

for(int i=1; i <= 5; i++)

list.addFirst(fullName[i]);

for(Person x : list)

System.out.print(x + " \n");

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

list.clear();

for(int i=0; i < 100; i++)

list.addFirst(fullName[i]);

for(int i=99; i >= 0; i--)

if(((Comparable<Person>)fullName[i]).compareTo(list.removeFirst()) != 0)

System.out.println("ERROR, elements don't match");

list.clear();

for(int i=0; i < 100; i++)

list.insert(fullName[i], (i+1));

for(int i=0; i <100; i++)

if(((Comparable<Person>)fullName[i]).compareTo(list.removeFirst()) != 0)

System.out.println("ERROR, elements don't match");

list.clear();

for(int i=0; i < 100; i++)

list.addLast(fullName[i]);

for(int i=1; i <= 100; i++)

if(((Comparable<Person>)fullName[i-1]).compareTo(list.removeFirst()) != 0)

System.out.println("ERROR, elements don't match");

if(list.size() != 0)

System.out.println("Size wrong, should be zero but is " + list.size());

list.clear();

for(int i=0; i < 100; i++)

list.addFirst(fullName[i]);

try {

list.insert(new Person("Foobar"), 102);

}

catch(RuntimeException e) {}

try {

list.insert(new Person("Foobar"), 0);

}

catch(RuntimeException e) {}

try {

if(list.locate(new Person("Foobar")) != -1)

System.out.println("ERROR, found an invalid element");

}

catch(RuntimeException e) {}

catch(Exception e) {

System.out.println("ERROR, failed with negative location "+e);

}

// did they use equals or == ?

list.addFirst(sentinal);

if(list.locate(new Person("Aaron Burr")) != 1)

System.out.println("ERROR, could not find valid element");

//===========

list2 = new LinearList<Integer>();

for(int i=1; i < 50; i++)

list2.addLast(new Integer(i));

// this will remove all the even numbers, only the odds print below.

for(int i=2; i < 25; i++)

list2.remove(i);

for(Integer x : list2)

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

System.out.println();

list2.clear();

for(int i=0; i < 10; i++) {

list2.addLast(50);

if(list2.removeLast() != 50) print("ERROR, 1");

}

if(list2.size() != 0) print("ERROR, 2");

list2.clear();

for(int i=0; i < 10; i++) {

list2.addFirst(75);

if(list2.removeLast() != 75) print("ERROR, 3");

}

list2.addLast(100);

if(list2.removeFirst() != 100) print("ERROR, 4");

list2.addLast(200);

if(list2.remove(1) != 200) print("ERROR, 5");

System.out.println("Size should be zero, is: " + list2.size());

for(Integer x : list2)

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

System.out.println();

for(int i=1; i <= 10; i++)

list2.insert(i,i);

for(Integer x : list2)

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

System.out.println();

for(int i=1; i <= 10; i++)

if(list2.get(i) != i) print("ERROR, 6. " +

" Looking for " + i + " but found " + list2.get(i));

for(int i=1; i <= 10; i++)

if(!list2.contains(i)) print("ERROR, 7");

long start = System.currentTimeMillis();

for(int i=0; i < 100000; i++)

list2.addLast(i);

long stop = System.currentTimeMillis();

System.out.println("AddLast time: " + (stop-start));

start = System.currentTimeMillis();

for(int i=0; i < 100000; i++)

list2.removeLast();

stop = System.currentTimeMillis();

System.out.println("removeLast time: " + (stop-start));

}

private void print(String s) {

System.out.println(s);

}

public static void main(String [] args) {

try {

new P2Grader();

}

catch(Exception e) {

System.out.println("ERROR: " + e);

e.printStackTrace();

}

}

// implements Comparable but NOT equals

class Person implements Comparable<Person> {

String name;

public Person(String n) {

name = n;

}

public String getName() {

return name;

}

public int compareTo(Person p) {

return name.compareTo(p.name);

}

public String toString() {

return name;

}

}

}