**Course**: Principals of Software Development – ENSF 409

**Lab 5**

**Instructor**: M. Moshirpour

**Student Name**: Mitchell Sawatzky

**Date Submitted**: Feb 23, 2016

Exercise B

Accessible.java

public interface Accessible {

public String getName();

public void setName(String newName);

}

Circle.java

class Circle extends Shape

{

private Double radius;

Circle(Double x\_origin, Double y\_origin, Double newradius, String name, Colour colour){

super(x\_origin, y\_origin, name, colour);

radius = newradius;

}

public Object clone() throws CloneNotSupportedException {

return super.clone();

}

public void set\_radius(Double newradius){

radius = newradius;

}

public Double get\_radius() {

return radius;

}

public Double area() {

return Math.PI \* Math.pow(radius, 2);

}

public Double perimeter() {

return 2 \* Math.PI \* radius;

}

public Double volume(){

return 0.0;

}

public String toString(){

String s = super.toString()+ "\nRadius: " + radius;

return s;

}

public void enlarge(double multiplier) throws SizeFactorException {

if (multiplier < LIMIT) {

throw new SizeFactorException(multiplier);

}

radius \*= multiplier;

}

public void shrink(double divisor) throws SizeFactorException {

if (divisor < LIMIT) {

throw new SizeFactorException(divisor);

}

radius /= divisor;

}

}

Colour.java

/\*

\* started by: M. Moussavi

\* Date: Feb 2015

\* Modified by: Mitchell Sawatzky

\*/

class Colour implements Cloneable

{

private String colour;

public Colour(String s) {

colour = new String(s);

}

public Object clone() throws CloneNotSupportedException {

return super.clone();

}

public void setColour(String newColour){

colour = newColour;

}

@Override

public String toString(){

return colour;

}

}

Geometry2.java

// import java.util.Iterator;

// import java.util.TreeSet;

/\*

\* started by: M. Moussavi

\* Date: Feb 2015

\* Modified by: Mitchell Sawatzky

\*/

public class Geometry2{

public static void main(String[] args) {

Rectangle r1 = new Rectangle(3.0, 4.0, 5.0, 6.0, "R1", new Colour("Black"));

Circle c1 = new Circle (13.0, 14.0, 15.0, "C1",new Colour ("Green"));

System.out.println("\nHere are the original values in r1:");

System.out.println(r1);

System.out.println("\nHere are the original values in c1:");

System.out.println(c1);

Rectangle r2 = new Rectangle(23.0, 24.0, 25.0, 26.0, "R2", new Colour("Black"));

Circle c2 = new Circle (33.0, 34.0, 35.0, "C2", new Colour("Yellow"));

System.out.println("\nHere are the original values in r2:");

System.out.println(r2);

System.out.println("Here are the original values in c2:");

System.out.println(c2);

Prism p1 = new Prism(43.0, 44.0, 45.0, 46.0, 47.0, "P1", new Colour("White"));

Prism p2 = new Prism (53.0, 54.0, 55.0, 56.0, 57.0, "P2", new Colour("Gray"));

System.out.println("\nHere are the original values in p1:");

System.out.println(p1);

System.out.println("\nHere are the original values in p2:");

System.out.println(p2);

// THE FOLLOWING CODE SEGMENT MUST BE UNCOMMENTED ONLY FOR EXERCISE A in Lab 5

// EXERCISE\_A\_BEGINS

// System.out.println("\n\nMaking r1 copy of r2, c1 copy of c2, p1 copy of p2:");

// try {

// r1 = (Rectangle)r2.clone();

// c1 = (Circle)c2.clone();

// p1 = (Prism)p2.clone();

// } catch (CloneNotSupportedException e) {

// System.out.println("Can't clone!");

// }

//

// r2.set\_length(1000.0);

// r2.getOrigin().setx(88.0);

// r2.getOrigin().sety(99.0);

// r2.name.setText("");

// c2.set\_radius(2000.00);

// c2.getOrigin().setx(188.0);

// c2.getOrigin().sety(199.0);

// c2.name.setText("");

// p2.set\_height(3000.0);

// p2.getOrigin().setx(88.0);

// p2.getOrigin().sety(99.0);

// p2.name.setText("");

//

// System.out.println("\nHere are values for r1 after trying to make it a copy of r2:");

// System.out.println(r1);

// System.out.println("\nHere are values for c1 after trying to make it a copy of c2:");

// System.out.println(c1);

// System.out.println("\nHere are values for p1 after trying to make it a copy of p2:");

// System.out.println(p1);

// EXERCISE\_A\_ENDS

// THE FOLLOWING CODE SEGMENT MUST BE UNCOMMENTED ONLY FOR EXERCISE B in Lab 5

// EXERCISE\_B\_BEGINS

try{

r1.enlarge(2.0);

r1.name.enlarge(3.0);

c1.shrink(2.0);

p1.enlarge(0.5);

} catch(SizeFactorException e){

System.out.println(e.getMessage());

}

System.out.println("\nHere are values for r1 after calling enlarge(2.0):");

System.out.println(r1);

System.out.println("\nHere is the font size for r1.name after calling enlarge(3.0):");

System.out.println(r1.name.getFontSize());

System.out.println("\nHere are values for c1 after calling shrink (2.0):");

System.out.println(c1);

System.out.println("\nHere are values for p1 after calling shrink (0.5):");

System.out.println(p1);

try{

p1.enlarge(0.5);

} catch(SizeFactorException e){

System.out.println(e.getMessage());

}

System.out.println("\nHere are values for p1 after calling shrink (0.5) -- UNCHANGED:");

System.out.println(p1);

// EXERCISE\_B\_ENDS

}

}

Point.java

/\*

\* started by: M. Moussavi

\* Date: Feb 2015

\* Modified by: Mitchell Sawatzky

\*/

class Point implements Cloneable

{

private Colour colour;

private Double xCoordinate, yCoordinate;

public Point(Double a, Double b, Colour c){

colour = (c);

xCoordinate = a;

yCoordinate = b;

}

public Object clone() throws CloneNotSupportedException {

Point obj = (Point)super.clone();

obj.colour = (Colour)colour.clone();

return obj;

}

@Override

public String toString() {

String s;

s = "X\_coordinate: " + xCoordinate + "\nY-coordinate: " + yCoordinate +

"\n" + colour + " point" ;

return s;

}

public Double getx() {

return xCoordinate;

}

void setx(Double newvalue){

xCoordinate = newvalue;

}

public Double gety() {

return yCoordinate;

}

public void sety(Double newvalue){

yCoordinate = newvalue;

}

public Double distance(Point other){

Double dist\_x = other.xCoordinate - xCoordinate;

Double dist\_y = other.yCoordinate - yCoordinate;

return (Math.sqrt(Math.pow(dist\_x, 2) + Math.pow(dist\_y, 2)));

}

static Double distance (Point that, Point other){

Double dist\_x = other.xCoordinate - that.xCoordinate;

Double dist\_y = other.yCoordinate - that.yCoordinate;

return (Math.sqrt(Math.pow(dist\_x, 2) + Math.pow(dist\_y, 2)));

}

}

Prism.java

class Prism extends Rectangle {

private Double height;

public Prism(Double x, Double y, Double l, Double w, Double h, String name, Colour colour)

{

super(x, y, l, w, name, colour);

height = h;

}

public Object clone() throws CloneNotSupportedException {

return super.clone();

}

public void set\_height(Double h)

{

height = h;

}

public Double height()

{

return height;

}

public Double area()

{

return 2 \* (length \* width) + 2 \* (height \* length) + 2 \* (height \* width);

}

public Double perimeter()

{

return width \* 2 + length \* 2;

}

public Double volume()

{

return width \* length \* height;

}

public String toString()

{

String s = super.toString()+ "\nheight: " + height;

return s;

}

public void enlarge(double multiplier) throws SizeFactorException {

if (multiplier < LIMIT) {

throw new SizeFactorException(multiplier);

}

height \*= multiplier;

super.enlarge(multiplier);

}

public void shrink(double divisor) throws SizeFactorException {

if (divisor < LIMIT) {

throw new SizeFactorException(divisor);

}

height /= divisor;

super.shrink(divisor);

}

}

Rectangle.java

/\*

\* started by: M. Moussavi

\* Date: Feb 2015

\* Modified by: Mitchell Sawatzky

\*/

class Rectangle extends Shape

{

protected Double width, length;

public Rectangle(Double x\_origin, Double y\_origin, Double newlength, Double newwidth, String name, Colour colour){

super(x\_origin, y\_origin, name, colour);

length= newlength;

width =newwidth;

}

public Object clone() throws CloneNotSupportedException {

return super.clone();

}

protected void set\_length(Double newlength){

length = newlength;

}

protected Double get\_length() {

return length;

}

protected Double area(){

return width \*length;

}

protected Double perimeter(){

return width \* 2 + length \* 2;

}

protected Double volume(){

return 0.0;

}

@Override

public String toString(){

String s = super.toString()+ "\nWidth: " + width + "\nLength: " + length;

return s;

}

public void enlarge(double multiplier) throws SizeFactorException {

if (multiplier < LIMIT) {

throw new SizeFactorException(multiplier);

}

width \*= multiplier;

length \*= multiplier;

}

public void shrink(double divisor) throws SizeFactorException {

if (divisor < LIMIT) {

throw new SizeFactorException(divisor);

}

width /= divisor;

length /= divisor;

}

}

Resizeable.java

public interface Resizeable {

static final double LIMIT = 1.0;

public void shrink(double divisor) throws SizeFactorException;

public void enlarge (double multiplier) throws SizeFactorException;

}

Shape.java

/\*

\* started by: M. Moussavi

\* Date: Feb 2015

\* Modified by: Mitchell Sawatzky

\*/

abstract class Shape implements Cloneable, Resizeable, Accessible

{

protected Point origin;

protected Text name;

abstract protected Double area();

abstract protected Double perimeter();

abstract protected Double volume();

protected Shape(Double x\_origin, Double y\_origin, String name, Colour colour){

origin = new Point(x\_origin,y\_origin, colour);

this.name = new Text(name);

}

protected Point getOrigin()

{

return origin;

}

public Object clone() throws CloneNotSupportedException {

Shape obj = (Shape)super.clone();

obj.origin = (Point)origin.clone();

obj.name = (Text)name.clone();

return obj;

}

protected Double distance( Shape other)

{

return origin.distance(other.origin);

}

protected Double distance( Shape a, Shape b)

{

return Point.distance(a.origin, b.origin);

}

protected void move(Double dx, Double dy)

{

origin.setx(origin.getx()+dx);

origin.sety(origin.gety()+dy);

}

@Override

public String toString(){

String s = "\nShape name: " + name + "\nOrigin: " + origin;

return s;

}

public String getName() {

return name.getText();

}

public void setName(String newName) {

name.setText(newName);

}

}

SizeFactorException.java

public class SizeFactorException extends Exception {

private static final long serialVersionUID = 9137726330394461024L;

public SizeFactorException(double n) {

super("Error: SizeFactorException: Resize factor " + n + " is less than 1.0");

}

}

Text.java

/\*

\* started by: M. Moussavi

\* Date: Feb 2015

\* Modified by: Mitchell Sawatzky

\*/

class Text implements Cloneable, Resizeable

{

private final Double DEFAULT\_SIZE = 10.0;

private Colour colour;

private Double fontSize;

private String text;

public Text(String text) {

this.text = text;

fontSize = DEFAULT\_SIZE;

}

public Object clone() throws CloneNotSupportedException {

Text obj = (Text)super.clone();

if (colour != null)

obj.colour = (Colour)colour.clone();

return obj;

}

public Double getFontSize(){

return fontSize;

}

public void setColour(String s){

colour = new Colour(s);

}

public void setText(String newText){

text = newText;

}

public String getText(){

return text ;

}

@Override

public String toString(){

return (text);

}

public void enlarge(double multiplier) throws SizeFactorException {

if (multiplier < LIMIT) {

throw new SizeFactorException(multiplier);

}

fontSize \*= multiplier;

}

public void shrink(double divisor) throws SizeFactorException {

if (divisor < LIMIT) {

throw new SizeFactorException(divisor);

}

fontSize /= divisor;

}

}

Terminal Output:

Mitchell@ttys001 17:26 {0} [5]$ java Geometry2

Here are the original values in r1:

Shape name: R1

Origin: X\_coordinate: 3.0

Y-coordinate: 4.0

Black point

Width: 6.0

Length: 5.0

Here are the original values in c1:

Shape name: C1

Origin: X\_coordinate: 13.0

Y-coordinate: 14.0

Green point

Radius: 15.0

Here are the original values in r2:

Shape name: R2

Origin: X\_coordinate: 23.0

Y-coordinate: 24.0

Black point

Width: 26.0

Length: 25.0

Here are the original values in c2:

Shape name: C2

Origin: X\_coordinate: 33.0

Y-coordinate: 34.0

Yellow point

Radius: 35.0

Here are the original values in p1:

Shape name: P1

Origin: X\_coordinate: 43.0

Y-coordinate: 44.0

White point

Width: 46.0

Length: 45.0

height: 47.0

Here are the original values in p2:

Shape name: P2

Origin: X\_coordinate: 53.0

Y-coordinate: 54.0

Gray point

Width: 56.0

Length: 55.0

height: 57.0

Error: SizeFactorException: Resize factor 0.5 is less than 1.0

Here are values for r1 after calling enlarge(2.0):

Shape name: R1

Origin: X\_coordinate: 3.0

Y-coordinate: 4.0

Black point

Width: 12.0

Length: 10.0

Here is the font size for r1.name after calling enlarge(3.0):

30.0

Here are values for c1 after calling shrink (2.0):

Shape name: C1

Origin: X\_coordinate: 13.0

Y-coordinate: 14.0

Green point

Radius: 7.5

Here are values for p1 after calling shrink (0.5):

Shape name: P1

Origin: X\_coordinate: 43.0

Y-coordinate: 44.0

White point

Width: 46.0

Length: 45.0

height: 47.0

Error: SizeFactorException: Resize factor 0.5 is less than 1.0

Here are values for p1 after calling shrink (0.5) -- UNCHANGED:

Shape name: P1

Origin: X\_coordinate: 43.0

Y-coordinate: 44.0

White point

Width: 46.0

Length: 45.0

height: 47.0

Exercise A

Date.java

class Date {

private int day, month, year;

public Date(int d, int m, int y)

{

day = d;

month = m;

year = y;

}

int get\_day() {

return day;

}

int get\_month() {

return month;

}

int get\_year() {

return year;

}

void set\_day(int d) {day = d;}

void set\_month(int m) {month = m;}

void set\_year(int y) {year = y;}

public String toString()

{

return day + "/" + month + "/" + year;

}

}

Demo.java

public class Demo {

public void lab5\_tests() {

Date d1 = new Date(2, 3, 1990);

Date d2 = new Date(2, 3, 1990);

Date d3 = new Date(2, 3, 1990);

Date d4 = new Date(2, 3, 1990);

LinkedList <Date> dates = new LinkedList < Date>();

dates.push\_back(1000, d1);

dates.push\_back(1001, d2);

dates.push\_back(1002, d3);

dates.push\_back(1003, d4);

System.out.println("\nPrinting list of dates just after its creation ...\n");

dates.print();

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

intlist.push\_back(2000, 23);

intlist.push\_back(2001, 24);

intlist.push\_back(2002, 266);

intlist.push\_back(2003, 323);

System.out.println("\nPrinting list of Integers just after its creation ...\n");

intlist.print();

LinkedList< Product> ltpr = new LinkedList< Product>();

if (ltpr.size() != 0){

System.out.println("\n1. Error: Incorrect size \n");

System.exit(1);

}

Product a = new Product ("Video Card", 2, 11, 1998, 33);

Product b = new Product ("Controller", 22, 10, 2008, 93);

Product c = new Product ("RAM", 31, 9, 2007, 3);

Product d = new Product ("Monitor", 2, 11, 1998, 83);

ltpr.push\_back(3000, a);

ltpr.push\_back(3001, b);

ltpr.push\_back(3002, c);

ltpr.push\_back(3003, d);

if(ltpr.size() != 4){

System.out.println("\n2. Error Incorrect size.\n");

System.exit(1);

}

System.out.println("\nPrinting list of products with 4 items ...\n");

ltpr.print();

ltpr.remove(3000);

ltpr.remove(3003);

if (ltpr.size() != 2){

System.out.println( "\n4. Error: Incorrect size. \n");

System.exit(1);

}

System.out.println( "\nPrinting list of products after two remove operations.\n");

ltpr.print();

System.out.println ("\nLet's look up some product names ...\n");

try\_to\_find(ltpr, 3002);

try\_to\_find(ltpr, 4000);

try\_to\_find(ltpr, 3001);

try\_to\_find(ltpr, 3000);

Point p1 = new Point(6, 8);

Point p2 = new Point(11, 34);

Point p3 = new Point(9, 109);

LinkedList<Point> ltp = new LinkedList< Point> ();

ltp.push\_back(5000,p1);

ltp.push\_back(5001, p2);

ltp.push\_back(5002,p3);

System.out.println("\nPrinting list of Points.\n");

ltp.print();

System.out.println ("\n\*\*\*----Finished testing----------------\*\*\*");

}

void print (LinkedList<?> lt)

{

if (lt.size() == 0)

System.out.println( " list is EMPTY.\n");

for (lt.go\_to\_first(); lt.cursor\_ok(); lt.step\_fwd()) {

System.out.println(lt);

}

}

public <T1> void try\_to\_find(LinkedList<T1> lt, Integer key )

{

lt.find(key);

if (lt.cursor\_ok())

System.out.println ("Found: " + lt );

else

System.out.println("Sorry, couldn't find key: " + key + " in the table.\n");

}

public static void main(String [] args)

{

Demo d = new Demo();

d.lab5\_tests();

}

}

LinkedList.java

class LinkedList<T1> {

private int sizeM;

private Node<T1> headM;

private Node<T1> cursorM;

public LinkedList()

{

sizeM = 0;

headM = null;

cursorM = null;

}

public int size()

{

return sizeM;

}

public boolean cursor\_ok()

{

return cursorM != null;

}

public Integer cursor\_key()

{

assert(cursor\_ok());

return cursorM.keyM;

}

public T1 cursor()

{

assert(cursor\_ok());

return cursorM.itemM;

}

public void push\_back(Integer keyA, T1 itemA){

Node<T1> new\_node = new Node<T1> (itemA, keyA, null );

if(headM == null)

headM = new\_node;

else {

cursorM = headM.nextM;

Node<T1> p = headM;

while (cursorM != null){

cursorM = cursorM.nextM;

p = p.nextM;

}

p.nextM = new\_node;

}

sizeM++;

}

public void insert (Integer keyA,T1 datumA)

{

if (headM == null || keyA.compareTo(headM.keyM) < 0)

{

Node<T1> new\_node = new Node<> (datumA,keyA, null);

headM = new\_node;

sizeM++;

}

else if (keyA.compareTo(headM.keyM) == 0) {

headM.itemM = datumA;

}

else {

Node<T1> before= headM;

Node<T1> after=headM.nextM;

while(after!= null && (keyA.compareTo(after.keyM)) > 0)

{

before=after;

after=after.nextM;

}

if(after!= null && keyA.compareTo(after.keyM) ==0)

{

after.itemM=datumA;

}

else

{

Node<T1> new\_node = new Node<>(datumA, keyA, null);

before.nextM = new\_node;

sizeM++;

}

}

}

void remove(Integer keyA )

{

if (headM == null || keyA.compareTo(headM.keyM) < 0)

return;

Node<T1> doomed\_node = null;

if (keyA.compareTo(headM.keyM) == 0) {

doomed\_node = headM;

headM = headM.nextM;

doomed\_node.nextM = null;

sizeM--;

}

else {

Node<T1> before = headM;

Node<T1> maybe\_doomed = headM.nextM;

while(maybe\_doomed != null && keyA.compareTo(maybe\_doomed.keyM) >0 ) {

before = maybe\_doomed;

maybe\_doomed = maybe\_doomed.nextM;

}

if (maybe\_doomed != null && (maybe\_doomed.keyM.compareTo(keyA)== 0)) {

// doomed\_node = maybe\_doomed;

before. nextM = maybe\_doomed.nextM;

maybe\_doomed = null;

sizeM--;

}

}

cursorM = null;

doomed\_node = null; // Does nothing if doomed\_node == 0.

}

void find(Integer keyA )

{

Node<T1> ptr=headM;

while (ptr!= null && (ptr.keyM.compareTo(keyA) >0 || ptr.keyM.compareTo(keyA) < 0))

{

ptr=ptr.nextM;

}

cursorM = ptr;

}

void go\_to\_first()

{

cursorM = headM;

}

void step\_fwd()

{

assert(cursor\_ok());

cursorM = cursorM . nextM;

}

void make\_empty()

{

headM = null;

sizeM = 0;

cursorM = null;

}

public void print()

{

cursorM = headM;

while (cursorM != null){

System.out.println("Key: " + cursorM.keyM + " || " + cursorM.itemM );

cursorM = cursorM.nextM;

}

}

public String toString()

{

String s;

if (cursor\_ok())

s = "Key: " + cursor\_key() + " || " + cursor();

else

s = "Not Found.";

return s;

}

}

Node.java

class Node<T1>

{

Integer keyM;

T1 itemM;

Node<T1> nextM;

public Node()

{

keyM = null;

itemM = null;

nextM = null;

}

public Node(T1 itemA, Integer keyA, Node<T1> nextA)

{

itemM= itemA ;

keyM = keyA;

nextM = nextA;

}

}

Point.java

class Point {

private double x\_coordinate, y\_coordinate;

static int counter = 0;

String id;

public Point(double a, double b)

{

x\_coordinate = a;

y\_coordinate = b;

id = "P" + ++counter;

}

public String toString()

{

String s;

s = "Point Id: " + id + "\nX\_coordinate: " + x\_coordinate + "\nY-coordinate: " + y\_coordinate;

return s;

}

public double getx()

{

return x\_coordinate;

}

void setx(double newvalue)

{

x\_coordinate = newvalue;

}

public double gety()

{

return y\_coordinate;

}

public void sety(double newvalue)

{

y\_coordinate = newvalue;

}

public double distance(Point other)

{

double dist\_x = other.x\_coordinate - x\_coordinate;

double dist\_y = other.y\_coordinate - y\_coordinate;

return (Math.sqrt(Math.pow(dist\_x, 2) + Math.pow(dist\_y, 2)));

}

static double distance (Point that, Point other)

{

double dist\_x = other.x\_coordinate - that.x\_coordinate;

double dist\_y = other.y\_coordinate - that.y\_coordinate;

return (Math.sqrt(Math.pow(dist\_x, 2) + Math.pow(dist\_y, 2)));

}

public static int count()

{

return counter;

}

public static void main(String [] args)

{

Point a = new Point (5, 6);

Point b = new Point (45, 69);

System.out.println(a.distance(b));

Point.distance(a, b);

System.out.println(a);

}

}

Product.java

class Product {

private String name;

private Date shelving;

private int shelf;

public Product(String n, int day, int month, int year, int sh){

name = n;

shelving = new Date(day, month, year);

shelf = sh;

}

public Date get\_date() {

return shelving;

}

public void set\_date(Date newDate) {

shelving = newDate;

}

public String get\_name() {

return name;

}

public void setname(String newName) {

name = newName;

}

public int get\_shelf() {

return shelf;

}

public void set\_shelf(int sh) {

shelf = sh;

}

public String toString(){

String s;

s = "Product Name: " + name + "||" + "Selving Date: " + shelving + "||" + "Shelf: " + shelf;

return s;

}

}

Terminal Output

Mitchell@ttys001 19:51 {0} [exC]$ java Demo

Printing list of dates just after its creation ...

Key: 1000 || 2/3/1990

Key: 1001 || 2/3/1990

Key: 1002 || 2/3/1990

Key: 1003 || 2/3/1990

Printing list of Integers just after its creation ...

Key: 2000 || 23

Key: 2001 || 24

Key: 2002 || 266

Key: 2003 || 323

Printing list of products with 4 items ...

Key: 3000 || Product Name: Video Card||Selving Date: 2/11/1998||Shelf: 33

Key: 3001 || Product Name: Controller||Selving Date: 22/10/2008||Shelf: 93

Key: 3002 || Product Name: RAM||Selving Date: 31/9/2007||Shelf: 3

Key: 3003 || Product Name: Monitor||Selving Date: 2/11/1998||Shelf: 83

Printing list of products after two remove operations.

Key: 3001 || Product Name: Controller||Selving Date: 22/10/2008||Shelf: 93

Key: 3002 || Product Name: RAM||Selving Date: 31/9/2007||Shelf: 3

Let's look up some product names ...

Found: Key: 3002 || Product Name: RAM||Selving Date: 31/9/2007||Shelf: 3

Sorry, couldn't find key: 4000 in the table.

Found: Key: 3001 || Product Name: Controller||Selving Date: 22/10/2008||Shelf: 93

Sorry, couldn't find key: 3000 in the table.

Printing list of Points.

Key: 5000 || Point Id: P1

X\_coordinate: 6.0

Y-coordinate: 8.0

Key: 5001 || Point Id: P2

X\_coordinate: 11.0

Y-coordinate: 34.0

Key: 5002 || Point Id: P3

X\_coordinate: 9.0

Y-coordinate: 109.0

\*\*\*----Finished testing----------------\*\*\*