Lab 9

Hannah Ebenezar 64011393
Kasita Sansanthad 64011426
Theint Nandar Su 64011752

Table of contents

Task 1

Bridge Pattern

Arc.java & Line.java

```java
public class Arc {

    public int x;
    public int y;
    public int r;

    public Arc(int x, int y, int r) {
        this.x = x;
        this.y = y;
        this.r = r;
    }

}
```

```java
public class Line {

    public int x1;
    public int y1;
    public int x2;
    public int y2;

    public Line(int x1, int y1, int x2, int y2) {
        this.x1 = x1;
        this.y1 = y1;
        this.x2 = x2;
        this.y2 = y2;
    }

}
```

Shape.java and its child class

```java
public abstract class Shape {

  protected int x;
  protected int y;

  protected DrawingService draw;

  public Shape(int x, int y) {
    this.x = x;
    this.y = y;
  }

  public abstract void draw();

  public void setDrawingService(DrawingService draw) {
    this.draw = draw;
  }

}
```

```java
public class Rectangle extends Shape {

  protected int x2;
  protected int y2;

  public Rectangle(int x, int y, int x2, int y2) {
    super(x, y);
    this.x2 = x2;
    this.y2 = y2;
  }

  public void draw() {
    draw.drawLine(x, y, x2, y);
    draw.drawLine(x2, y, x2, y2);
    draw.drawLine(x2, y2, x, y2);
    draw.drawLine(x, y2, x, y);
  }

}
```

```java
public class Circle extends Shape {

  protected int r;

  public Circle(int x, int y, int r) {
    super(x, y);
    this.r = r;


  blic void draw() {
    draw.drawCircle(x, y, r);
```

```java
public class Polygon extends Shape {
  private int[] xPoints;
  private int[] yPoints;
  private int nPoints;

  public Polygon(int[] xPoints, int[] yPoints, int nPoints) {
      super(x:0, y:0); // Assuming you don't need x, y for Polygon itself
      this.xPoints = xPoints;
      this.yPoints = yPoints;
      this.nPoints = nPoints;
  }

  @Override
  public void draw() {
      draw.drawPolygon(xPoints, yPoints, nPoints);
  }
}
```

DrawingService.java

```java
public abstract class DrawingService {

  public abstract void drawLine(int x1, int y1, int x2, int y2);
  public abstract void drawCircle(int x, int y, int r);
  public abstract void drawPolygon(int[] x, int[] y,int n);

}
```

Drawing Service Concrete Class

```java
public class WrapMonitor extends DrawingService {
  private Monitor monitor;

  public WrapMonitor() {
    monitor = Monitor.getMonitor();
  }

  public void drawLine(int x1, int y1, int x2, int y2) {
    monitor.draw_a_line(x1, y1, x2, y2);
  }

  public void drawCircle(int x, int y, int r) {
    monitor.draw_a_circle(x, y, r);
  }

  public void drawPolygon(int[] xPoints, int[] yPoints, int nPoints){
    monitor.draw_a_polygon(xPoints, yPoints, nPoints);
  }
}
```

```java
public class Monitor extends JComponent {

    private static Monitor monitor;
    private List<Arc> arcs;
    private List<Line> lines;
    private List<Polygon> polygons;
    private JFrame frame;

    public static Monitor getMonitor() {
        if (monitor == null) {
            monitor = new Monitor();
        }
        return monitor;
    }

    private Monitor() {
        super();
        frame = new JFrame(title:"Monitor");
        frame.setSize(width:400, height:400);
```

```java
public class WrapXMLWriter extends DrawingService{
    private XMLWriter writer;

    public WrapXMLWriter(){
      writer =  XMLWriter.getWriter();
    }

    public void drawLine(int x1, int y1, int x2, int y2) {
        writer.writeLine(x1, x2, y1, y2);
    }

    public void drawCircle(int x, int y, int r) {
        writer.writeCircle(x, y, r);
    }

    public void drawPolygon(int[] x, int[] y, int z){
        writer.writePolygon(x, y, z);
    }
```

```java
public class Printer extends JComponent {

    private static Printer printer;

    public static Printer getPrinter() {
        if (printer == null) {
            printer = new Printer();
        }
        return printer;
    }

    private List<Arc> arcs;
    private List<Line> lines;
    private List<Polygon> polygons;

    private JFrame frame;

    private Printer() {
```

```java
public class WrapPrinter extends DrawingService {
  private Printer printer;

  public WrapPrinter() {
    printer = Printer.getPrinter();
  }

  public void drawLine(int x1, int y1, int x2, int y2) {
    printer.drawLine(x1, x2, y1, y2);
  }

  public void drawCircle(int x, int y, int r) {
    printer.drawCircle(x, y, r);
  }

  public void drawPolygon(int[] x, int[] y, int z){
    printer.drawPolygon(x, y, z);
  }
}
```

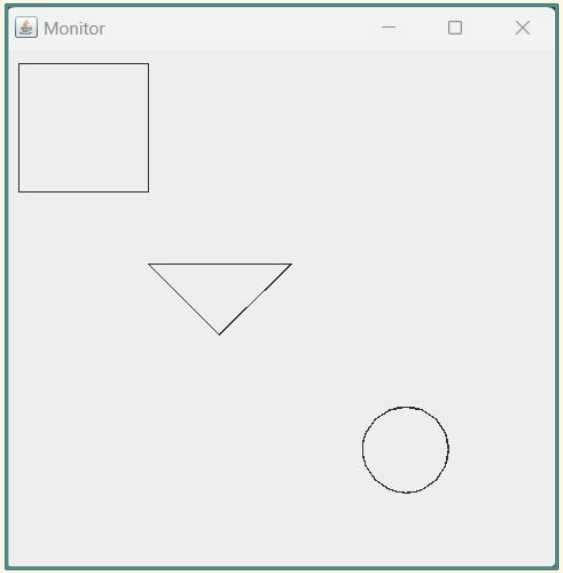
```java
public class XMLWriter{
    private static XMLWriter writer;

    public static XMLWriter getWriter() {
        if (writer == null) {
            writer = new XMLWriter();
        }
        return writer;
    }


    public void writeLine(int x1, int y1, int x2, int y2) {
        String line = String.format(format:"<line x1=\"%d\" y
        System.out.println(line);
    }
```

Testing

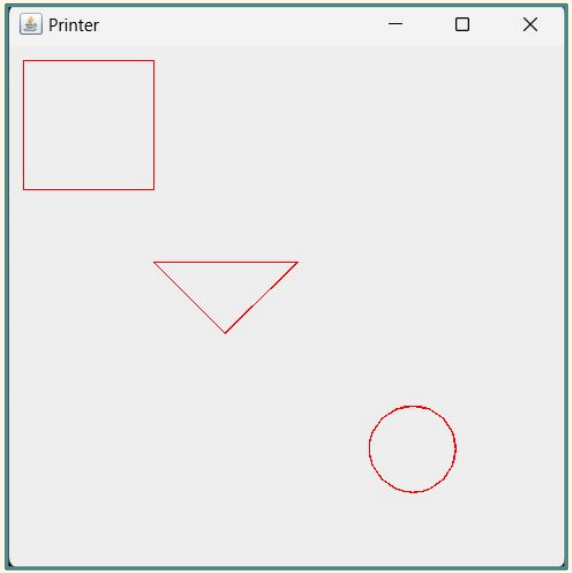
```java
public class Test {

  Run | Debug
  public static void main(String[] args) throws InterruptedException {

    List<Shape> shapes = new LinkedList<Shape>();

    shapes.add(new Rectangle(x:10, y:10, x2:100, y2:100));
    shapes.add(new Circle(x:250, y:250, r:30));
    shapes.add(new Polygon(new int[]{100, 150, 200}, new int[]{150, 200, 150}, nPoints:3));

    for (Shape s : shapes) {
      s.setDrawingService(new WrapMonitor());
    }

    for (Shape s : shapes) {
      s.draw();
    }

    Thread.sleep(millis:2000);

    for (Shape s : shapes) {
      s.setDrawingService(new WrapPrinter());
    }

    Thread.sleep(millis:2000);

    for (Shape s : shapes) {
      s.draw();
    }

    Thread.sleep(millis:2000);

    for (Shape s : shapes) {
      s.setDrawingService(new WrapXMLWriter());
    }

    Thread.sleep(millis:2000);

    for (Shape s : shapes) {
      s.draw();
    }
```

Running

Monitor



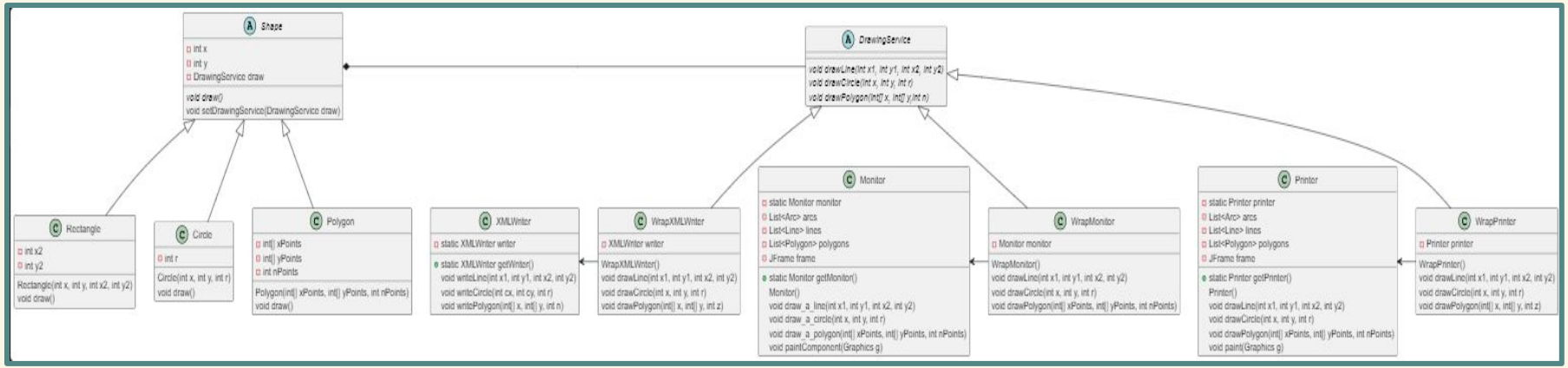
Printer



XML

```
<line x1="10" y1="100" x2="10" y2="10"/>

<line x1="100" y1="100" x2="10" y2="100"/>

<line x1="100" y1="10" x2="100" y2="100"/>

Mouse moved: (368, 80)
<line x1="10" y1="10" x2="100" y2="10"/>

<circle cx="250" cy="250" r="30"/>

<polygon points="(100,150)(150,200)(200,150"/>
```

Class Diagram



Task 2

Mediator Pattern

Buyer Classes

```java
public class Buyer {

    Mediator mediator;
    String unitOfCurrency;

    public Buyer(Mediator mediator, String unitOfCurrency) {
        this.mediator = mediator;
        this.unitOfCurrency = unitOfCurrency;
    }

    public boolean attemptToPurchase(float bid) {
        System.out.println("Buyer attempting a bid of " + bid + " " + unitOfCurrency);
        return mediator.placeBid(bid, unitOfCurrency);
    }
}
```

```java
public class SwedishBuyer extends Buyer {

    public SwedishBuyer(Mediator mediator) {
        super(mediator, unitOfCurrency:"krona");
        this.mediator.registerSwedishBuyer(this);
    }
}
```

```java
public class FrenchBuyer extends Buyer {

    public FrenchBuyer(Mediator mediator) {
        super(mediator, unitOfCurrency:"euro");
        this.mediator.registerFrenchBuyer(this);
    }
}
```

Dollar Converter Class

```java
public class DollarConverter {

    Mediator mediator;

    public static final float DOLLAR_UNIT = 1.0f;
    public static final float EURO_UNIT = 0.7f;
    public static final float KRONA_UNIT = 8.0f;

    public DollarConverter(Mediator mediator) {
        this.mediator = mediator;
        mediator.registerDollarConverter(this);
    }

    private float convertEurosToDollars(float euros) {
        float dollars = euros * (DOLLAR_UNIT / EURO_UNIT);
        System.out.println("Converting " + euros + " euros to " + dollars + " dollars");
        return dollars;
    }

    private float convertKronorToDollars(float kronor) {
        float dollars = kronor * (DOLLAR_UNIT / KRONA_UNIT);
        System.out.println("Converting " + kronor + " kronor to " + dollars + " dollars");
        return dollars;
    }

    public float convertCurrencyToDollars(float amount, String unitOfCurrency) {
        if ("krona".equalsIgnoreCase(unitOfCurrency)) {
            return convertKronorToDollars(amount);
        } else {
            return convertEurosToDollars(amount);
        }
    }
}
```

American Seller Class

```
public class AmericanSeller {

    Mediator mediator;
    float priceInDollars;

    public AmericanSeller(Mediator mediator, float priceInDollars) {
        this.mediator = mediator;
        this.priceInDollars = priceInDollars;
        this.mediator.registerAmericanSeller(this);
    }

    public boolean isBidAccepted(float bidInDollars) {
        if (bidInDollars >= priceInDollars) {
            System.out.println("Seller accepts the bid of " + bidInDollars + " dollars\n");
            return true;
        } else {
            System.out.println("Seller rejects the bid of " + bidInDollars + " dollars\n");
            return false;
        }
    }

}
```

Mediator Class

```
public class Mediator {

    private AmericanSeller americanSeller;
    private SwedishBuyer swedishBuyer;
    private FrenchBuyer frenchBuyer;
    private DollarConverter dollarConverter;

    public void registerAmericanSeller(AmericanSeller americanSeller) {
        this.americanSeller = americanSeller;
    }

    public void registerSwedishBuyer(SwedishBuyer swedishBuyer) {
        this.swedishBuyer = swedishBuyer;
    }

    public void registerFrenchBuyer(FrenchBuyer frenchBuyer) {
        this.frenchBuyer = frenchBuyer;
    }

    public void registerDollarConverter(DollarConverter dollarConverter) {
        this.dollarConverter = dollarConverter;
    }

    public boolean placeBid(float bid, String unitOfCurrency) {
        float bidInDollars = dollarConverter.convertCurrencyToDollars(bid, unitOfCurrency);
        return americanSeller.isBidAccepted(bidInDollars);
    }
}
```

Mediator class

central point for communication between various objects.

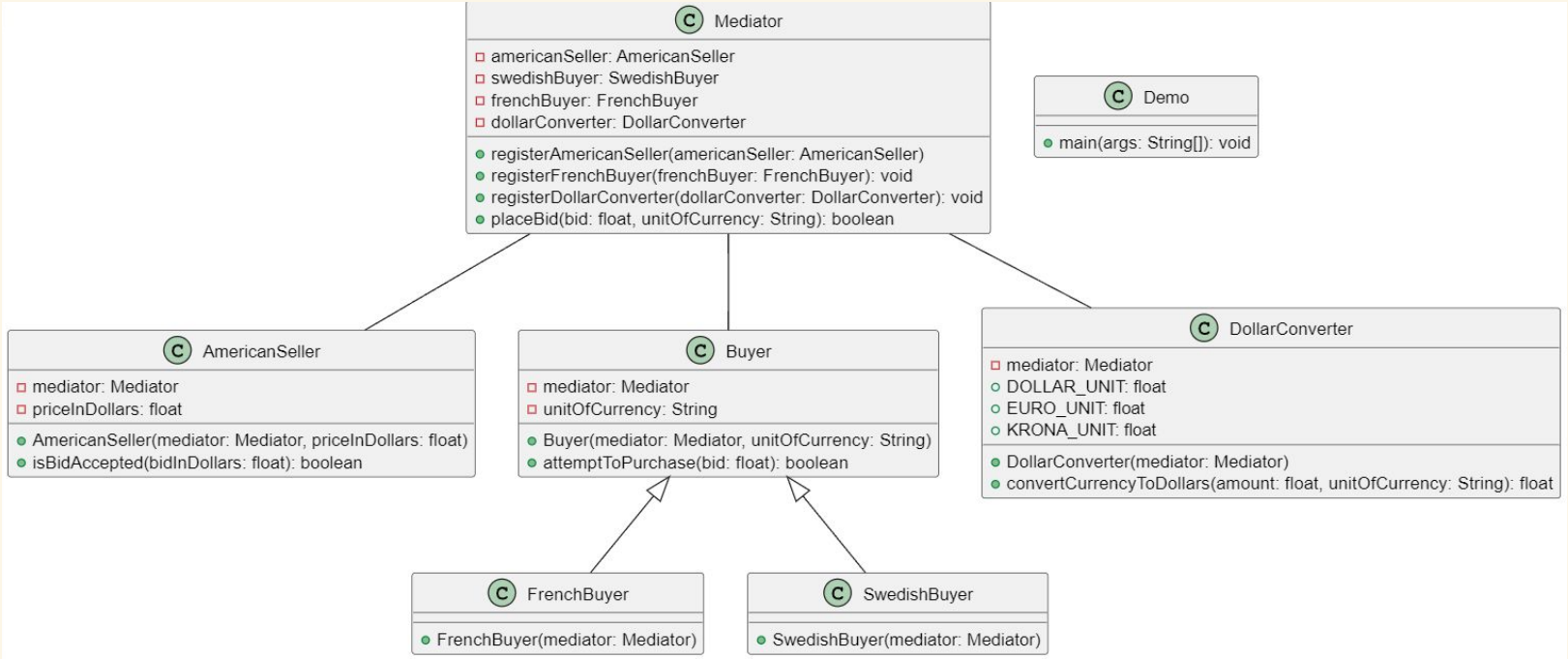
Testing and Running

```java
public class Demo {

    Run | Debug
    public static void main(String[] args) {

        Mediator mediator = new Mediator();

        Buyer swedishBuyer = new SwedishBuyer(mediator);
        Buyer frenchBuyer = new FrenchBuyer(mediator);
        float sellingPriceInDollars = 10.0f;
        AmericanSeller americanSeller = new AmericanSeller(mediator, sellingPriceInDollars);
        DollarConverter dollarConverter = new DollarConverter(mediator);

        float swedishBidInKronor = 55.0f;
        while (!swedishBuyer.attemptToPurchase(swedishBidInKronor)) {
            swedishBidInKronor += 15.0f;
        }

        float frenchBidInEuros = 3.0f;
        while (!frenchBuyer.attemptToPurchase(frenchBidInEuros)) {
            frenchBidInEuros += 1.5f;
        }

    }
}
```

Output

```
Buyer attempting a bid of 55.0 krona
Converting 55.0 kronor to 6.875 dollars
Seller rejects the bid of 6.875 dollars

Buyer attempting a bid of 70.0 krona
Converting 70.0 kronor to 8.75 dollars
Seller rejects the bid of 8.75 dollars

Buyer attempting a bid of 85.0 krona
Converting 85.0 kronor to 10.625 dollars
Seller accepts the bid of 10.625 dollars

Buyer attempting a bid of 3.0 euro
Converting 3.0 euros to 4.285714 dollars
Seller rejects the bid of 4.285714 dollars

Buyer attempting a bid of 4.5 euro
Converting 4.5 euros to 6.4285717 dollars
Seller rejects the bid of 6.4285717 dollars

Buyer attempting a bid of 6.0 euro
Converting 6.0 euros to 8.571428 dollars
Seller rejects the bid of 8.571428 dollars

Buyer attempting a bid of 7.5 euro
Converting 7.5 euros to 10.714286 dollars
Seller accepts the bid of 10.714286 dollars
```

Class Diagram with PlantUML



Task 3

Proxy Pattern

AbstractMap.java

```java
public interface AbstractMap {

    public String find(String key) throws Exception;
    public void add(String key, String value) throws Exception;

}
```

Interface

Map and MapProxy will implements AbstractMap interface

Map.java

Map is a real subject that MapProxy Object represents.

```
import java.io.*;
import java.util.*;

public class Map implements AbstractMap {
    private String fileName;
    private final String header =
    " -- Generated file, do not edit --";

    public Map(String fileName)
    {
        this.fileName = fileName;
        File file = new File(fileName);
        // Ensure that the file exists.
        try {
            file.createNewFile();
        } catch (IOException e) {
            e.printStackTrace();
            System.exit(status:1);
        }
    }

    public String find(String key) throws IOException
    {
        // Open the file, look up the value of the key
        // given, then close it.
        InputStream is = new FileInputStream(fileName);
        Properties props = new Properties();
        props.load(is);
        is.close();
        return props.getProperty(key);
    }
```

```
    public void add(String key, String value) throws IOException
    {
        // Open the file, look up the value of the key
        // given, then close it.
        InputStream is = new FileInputStream(fileName);
        Properties props = new Properties();
        props.load(is);
        is.close();
        props.setProperty(key, value);
        OutputStream os =new FileOutputStream(fileName);
        props.store(os, header);
    }

}
```

MapProxy.java

```java
import java.util.HashMap;

public class MapProxy implements AbstractMap {

    public MapProxy(String fileName) {
        this.fileName = fileName;
    }

    @Override
    public String find(String key) throws Exception {
        return getMap().find(key);
    }

    @Override
    public void add(String key, String value) throws Exception {
        getMap().add(key, value);
    }

    private Map getMap()
    {
    if (map == null) {
        map = new Map(fileName);
    } // end of if (map == null)
    return map;
    }
```

```java
    private String get(String key)
    {
    return (String) hashtable.get(key);
    }

    private void put(String key, String value)
    {
    hashtable.put(key, value);
    }

    private String fileName;
    private Map map = null;
    private HashMap hashtable = new HashMap();
}
```

Testing and Running (MapProxyTest.py)

```
import junit.framework.*;
import java.util.Date;

public class MapProxyTest extends TestCase {

    private String fileName;
    private final int COUNT = 10000;

    public MapProxyTest(String name) {
    super(name);
    }

    Run | Debug
    public static void main (String[] args) {
    junit.textui.TestRunner.run(suite());
    }

    public static Test suite() {
    TestSuite suite = new TestSuite(MapProxyTest.class);
    return suite;
    }

    public void setUp() throws Exception {
    String sep = System.getProperty(key:"file.separator");
    if (sep.equals(anObject:"/")) {
        // assume we are on Unix.
        fileName = "/tmp/key_values";
    } // end of if (sep.equals("/"))
    else {
        fileName = "C:\\TEMP\\key_values";
    } // end of else
    }

    public void test1() throws Exception {
    AbstractMap map = new MapProxy(fileName);
    }
```

```
public void test1() throws Exception {
AbstractMap map = new MapProxy(fileName);
}

public void test2() throws Exception {
String value = "Eric Dubuis";
AbstractMap map = new MapProxy(fileName);
map.add(key:"name", value);
String result = map.find(key:"name");
assertEquals(value, result);
}

public void test3() throws Exception {
// Performance test.
String value = "Eric Dubuis";
AbstractMap map = new MapProxy(fileName);
map.add(key:"name", value);
System.out.println("\nStarting loop at: " + new Date().toString());
for (int i = 0; i < COUNT; i++) {
    map.find(key:"name");
} // end of for (int i = 0; i < COUNT; i++)
System.out.println("Finished loop at: " + new Date().toString());
}

public void tearDown() throws Exception {
//
}
```

Output:

```
kasitasansanthad@Kasitas-MacBook-Pro lab9task3 %  /usr/bin/env /Users/kasitasansanthad/g/04tp901j63nc_nzw9dsbcwqw0000gn/T/cp_16vr3vuhrbts7mwqxfyq95vj4.argfile MapProxyTest
...
Starting loop at: Wed Feb 21 22:38:20 ICT 2024
Finished loop at: Wed Feb 21 22:38:20 ICT 2024

Time: 0.12

OK (3 tests)

<ownloads/jdk-21.0.1.jdk/Contents/Home/bin/java @/var/folders/1g/04tp901j63nc_nzw9dsbc
...
Starting loop at: Wed Feb 21 23:09:50 ICT 2024
Finished loop at: Wed Feb 21 23:09:50 ICT 2024

Time: 0.153

OK (3 tests)

kasitasansanthad@Kasitas-MacBook-Pro lab9task3 %
```

Testing and Running (MapTest.java)

```java
import junit.framework.*;
import java.util.Date;

public class MapTest extends TestCase {

    private String fileName;
    private final int COUNT = 10000;

    public MapTest(String name) {
    super(name);
    }

    Run | Debug
    public static void main (String[] args) {
    junit.textui.TestRunner.run(suite());
    }

    public static Test suite() {
    TestSuite suite = new TestSuite(MapTest.class);
    return suite;
    }

    public void setUp() throws Exception {
    String sep = System.getProperty(key:"file.separator");
    if (sep.equals(anObject:"/")) {
        // assume we are on Unix.
        fileName = "/tmp/key_values";
    } // end of if (sep.equals("/"))
    else {
        fileName = "C:\\TEMP\\key_values";
    } // end of else
    }

    public void test1() throws Exception {
    AbstractMap map = new Map(fileName);
    }
```

```java
public void test2() throws Exception {
String value = "Eric Dubuis";
AbstractMap map = new Map(fileName);
map.add(key:"name", value);
String result = map.find(key:"name");
assertEquals(value, result);
}

public void test3() throws Exception {
// Performance test.
String value = "Eric Dubuis";
AbstractMap map = new Map(fileName);
map.add(key:"name", value);
System.out.println("\nStarting loop at: " + new Date().toString());
for (int i = 0; i < COUNT; i++) {
    map.find(key:"name");
} // end of for (int i = 0; i < COUNT; i++)
System.out.println("Finished loop at: " + new Date().toString());
}

public void tearDown() throws Exception {
//
}
```

Output:

```
kasitasansanthad@Kasitas-MacBook-Pro lab9task3 %  cd /Users/kasitasansant
Downloads/jdk-21.0.1.jdk/Contents/Home/bin/java @/var/folders/1g/04tp901j
...
Starting loop at: Wed Feb 21 22:38:16 ICT 2024
Finished loop at: Wed Feb 21 22:38:16 ICT 2024

Time: 0.12

OK (3 tests)

<ownloads/jdk-21.0.1.jdk/Contents/Home/bin/java @/var/folders/1g/04tp901j
...
Starting loop at: Wed Feb 21 23:11:18 ICT 2024
Finished loop at: Wed Feb 21 23:11:19 ICT 2024

Time: 0.118

OK (3 tests)

kasitasansanthad@Kasitas-MacBook-Pro lab9task3 %
```

Testing and Running (Client.java)

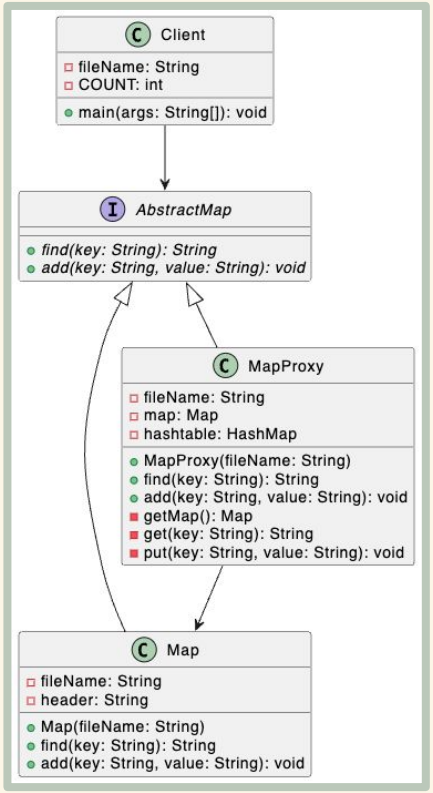
```java
import java.util.Date;

public class Client {

    private static String fileName;
    private final static int COUNT = 10000;

    Run | Debug
    public static void main (String[] args) {

    // Determine file name for properties file.
    String sep = System.getProperty(key:"file.separator");
    if (sep.equals(anObject:"/")) {
        // assume we are on Unix.
        fileName = "/tmp/key_values";
    } // end of if (sep.equals("/"))
    else {
        // Assume we are on Windows.
        fileName = "C:\\TEMP\\key_values";
    } // end of else

    System.out.println
        ("Notice that file "+fileName+" is used to store key-

    // Performance test.
    AbstractMap map;
    String key = "name";
    String value = "Eric Dubuis";

    // Testing raw access to properties file.
    System.out.println("Accessing "+COUNT+" times a Map objec
```

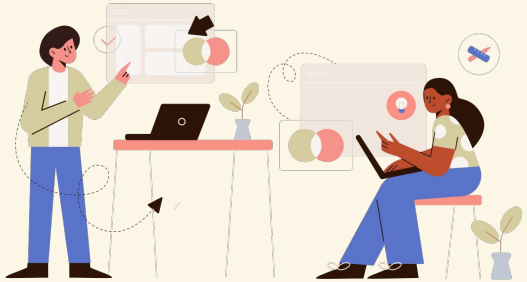
```java
    try {
        map = new Map(fileName);
        map.add(key:"name", value);
        System.out.println("  Starting loop at: " + new Date().toString());
        for (int i = 0; i < COUNT; i++) {
        map.find(key:"name");
        } // end of for (int i = 0; i < COUNT; i++)
        System.out.println("  Finished loop at: " + new Date().toString()+"\n");

        // Testing raw access to properties file.
        System.out.println("Accessing "+COUNT+" times a Map proxy object:");

        map = new MapProxy(fileName);
        map.add(key:"name", value);
        System.out.println("  Starting loop at: " + new Date().toString());
        for (int i = 0; i < COUNT; i++) {
        map.find(key:"name");
        } // end of for (int i = 0; i < COUNT; i++)
        System.out.println("  Finished loop at: " + new Date().toString());

    } catch (Exception e) {
        e.printStackTrace();
    }

    } // end of main ()
}
```

Output:

```
kasitasansanthad@Kasitas-MacBook-Pro lab9task3 %  cd /Users/kas
Downloads/jdk-21.0.1.jdk/Contents/Home/bin/java @/var/folders/
Notice that file /tmp/key_values is used to store key-value pa

Accessing 10000 times a Map object:
  Starting loop at: Wed Feb 21 23:04:06 ICT 2024
  Finished loop at: Wed Feb 21 23:04:06 ICT 2024

Accessing 10000 times a Map proxy object:
  Starting loop at: Wed Feb 21 23:04:06 ICT 2024
  Finished loop at: Wed Feb 21 23:04:06 ICT 2024
kasitasansanthad@Kasitas-MacBook-Pro lab9task3 %
```

Class diagram





Thanks!

Do you have any questions?
youremail@freepik.com
+34 654 321 432
yourwebsite.com