1.

int[] data = {27, 51, 33, -1, 101};

2.

data[0] = 3

data[1] = 3

data[2] = 0

data[3] = 0

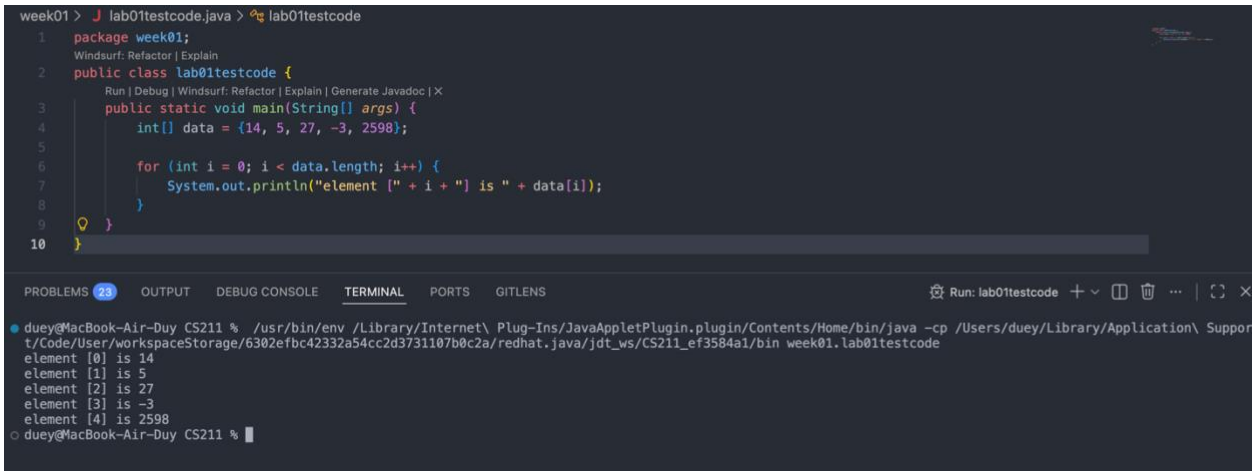
data[4] = 6

data[5] = 9

data[6] = 0

data[7] = -18

3.


week01 > lab01testcode.java > lab01testcode
Windsurf: Refactor | Explain
Run | Debug | Windsurf: Refactor | Explain | Generate Javadoc | ×
PROBLEMS 23
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
GITLENS
Run: lab01testcode

```java
package week01;
public class lab01testcode {
    public static void main(String[] args) {
        int[] data = {14, 5, 27, -3, 2598};

        for (int i = 0; i < data.length; i++) {
            System.out.println("element [" + i + "] is " + data[i]);
        }
    }
}
```

```
duey@MacBook-Air-Duy CS211 %  /usr/bin/env /Library/Internet\ Plug-Ins/JavaAppletPlugin.plugin/Contents/Home/bin/java -cp /Users/duey/Library/Application\ Suppor
t/Code/User/workspaceStorage/6302efbc42332a54cc2d3731107b0c2a/redhat.java/jdt_ws/CS211_ef3584a1/bin week01.lab01testcode
element [0] is 14
element [1] is 5
element [2] is 27
element [3] is -3
element [4] is 2598
duey@MacBook-Air-Duy CS211 %
```

4.

2 [0,0,1,0]

1 [0,0,1,0]

3 [0,0,1,1]

2 [0,0,1,1]

5.

a1[0] = 1

a1[1] = 3

a1[2] = -3

a1[3] = 13

a1[4] = -4

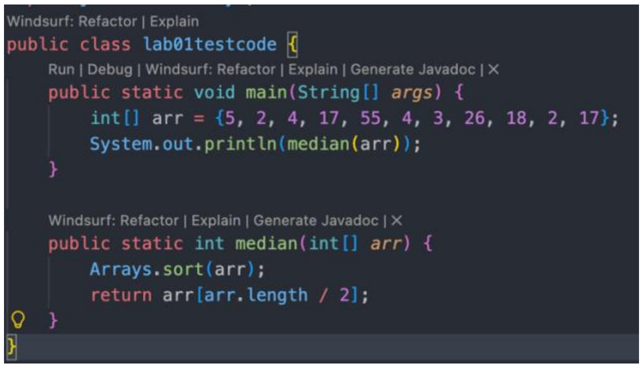
a1[5] = -24

a1[6] = -6

a1[7] = -14

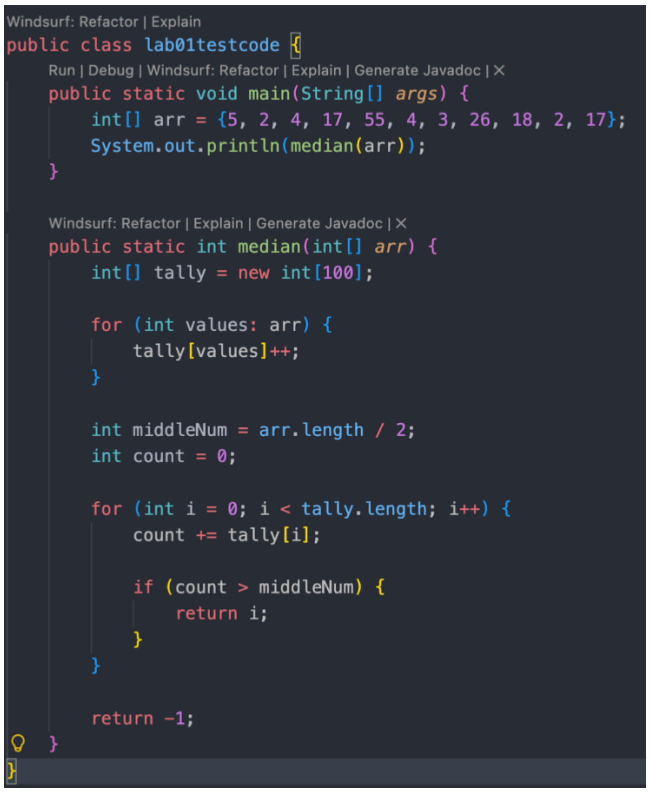
6.

Arrays.sort method:



```
Windsurf: Refactor | Explain
public class lab01testcode {
    Run | Debug | Windsurf: Refactor | Explain | Generate Javadoc | ×
    public static void main(String[] args) {
        int[] arr = {5, 2, 4, 17, 55, 4, 3, 26, 18, 2, 17};
        System.out.println(median(arr));
    }

    Windsurf: Refactor | Explain | Generate Javadoc | ×
    public static int median(int[] arr) {
        Arrays.sort(arr);
        return arr[arr.length / 2];
    }
}
```

tally technique:

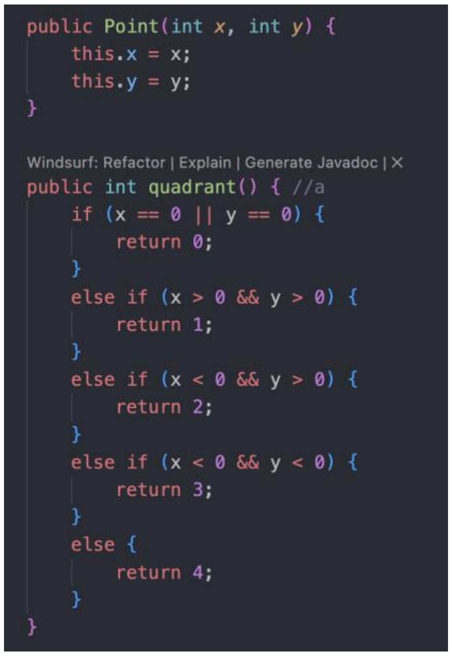

Windsurf: Refactor | Explain

```java
public class lab01testcode {
    public static void main(String[] args) {
        int[] arr = {5, 2, 4, 17, 55, 4, 3, 26, 18, 2, 17};
        System.out.println(median(arr));
    }

    public static int median(int[] arr) {
        int[] tally = new int[100];

        for (int values: arr) {
            tally[values]++;
        }

        int middleNum = arr.length / 2;
        int count = 0;

        for (int i = 0; i < tally.length; i++) {
            count += tally[i];

            if (count > middleNum) {
                return i;
            }
        }

        return -1;
    }
}
```

Run | Debug | Windsurf: Refactor | Explain | Generate Javadoc | ✕
Windsurf: Refactor | Explain | Generate Javadoc | ✕

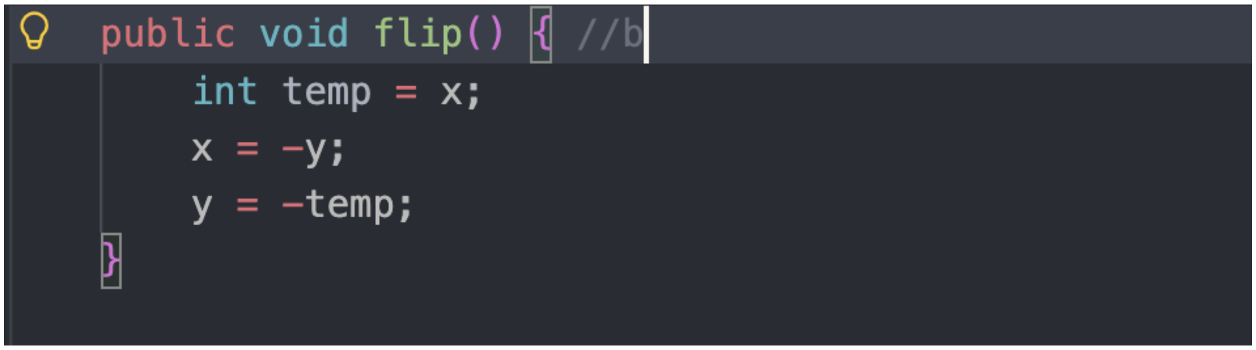
7.

A.



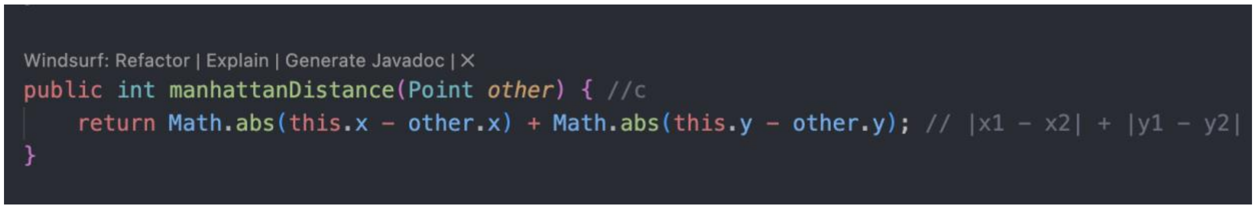
```java
public Point(int x, int y) {
    this.x = x;
    this.y = y;
}

public int quadrant() { //a
    if (x == 0 || y == 0) {
        return 0;
    }
    else if (x > 0 && y > 0) {
        return 1;
    }
    else if (x < 0 && y > 0) {
        return 2;
    }
    else if (x < 0 && y < 0) {
        return 3;
    }
    else {
        return 4;
    }
}
```

B.



```java
public void flip() { //b
        int temp = x;
        x = -y;
        y = -temp;
    }
```

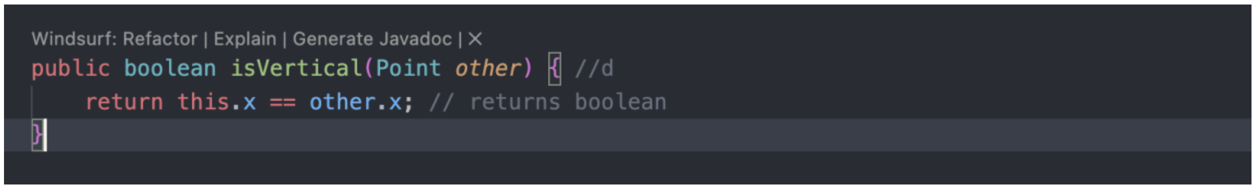
C.



Windsurf: Refactor | Explain | Generate Javadoc | ✕

```java
public int manhattanDistance(Point other) { //c
    return Math.abs(this.x - other.x) + Math.abs(this.y - other.y); // |x1 - x2| + |y1 - y2|
}
```

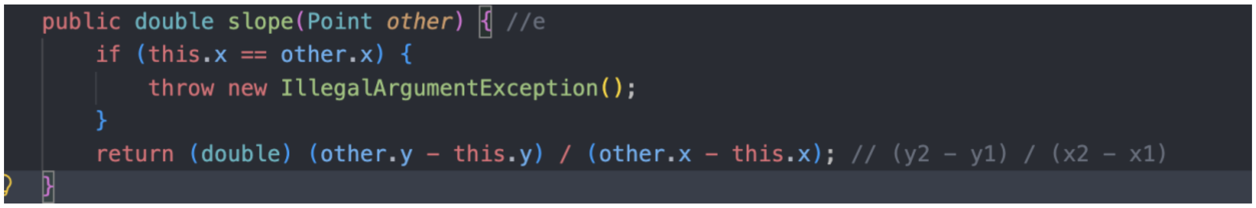
D.



Windsurf: Refactor | Explain | Generate Javadoc | ✕

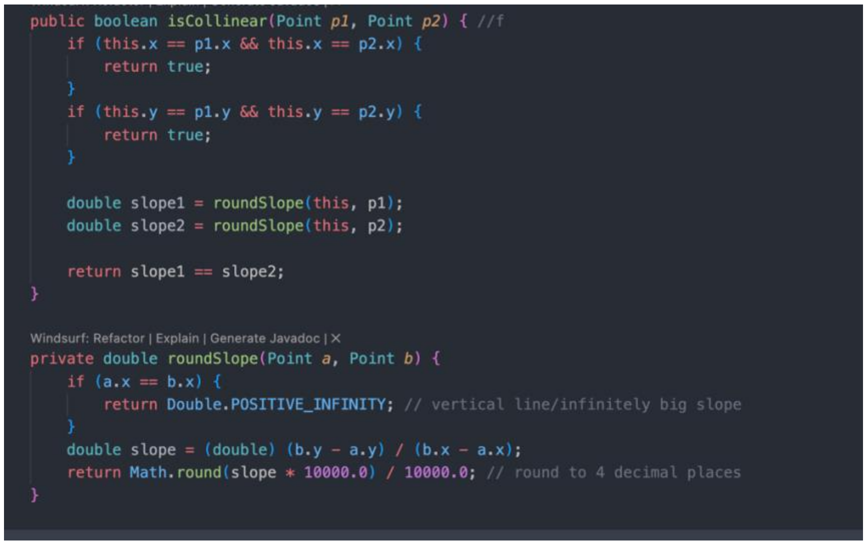
```java
public boolean isVertical(Point other) { //d
    return this.x == other.x; // returns boolean
}
```

E.



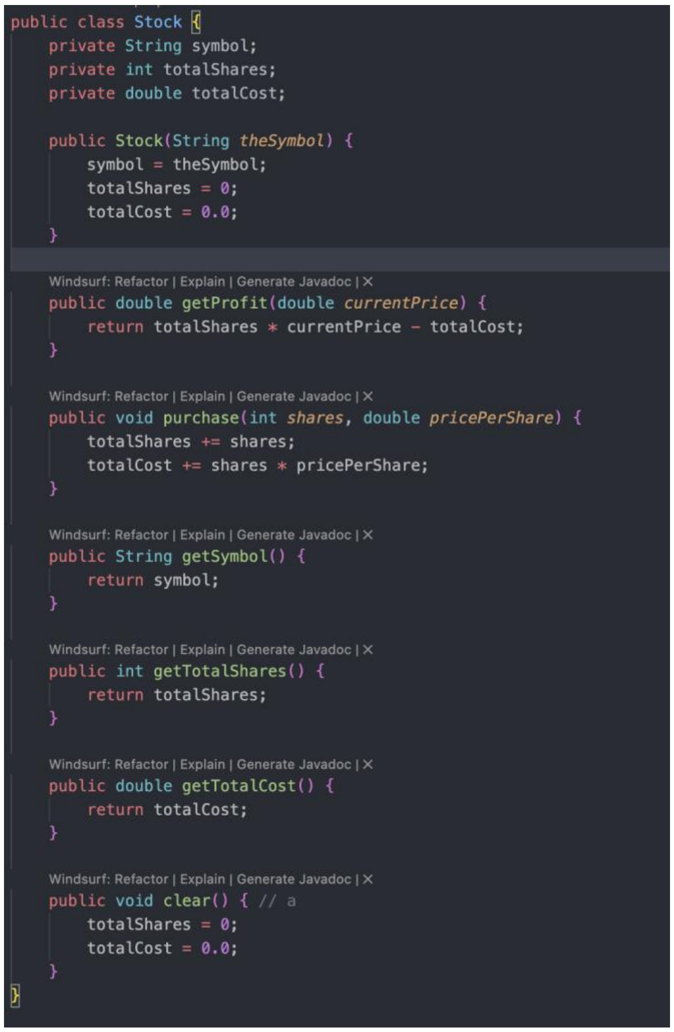
```java
public double slope(Point other) { //e
    if (this.x == other.x) {
        throw new IllegalArgumentException();
    }
    return (double) (other.y - this.y) / (other.x - this.x); // (y2 - y1) / (x2 - x1)
}
```

F.

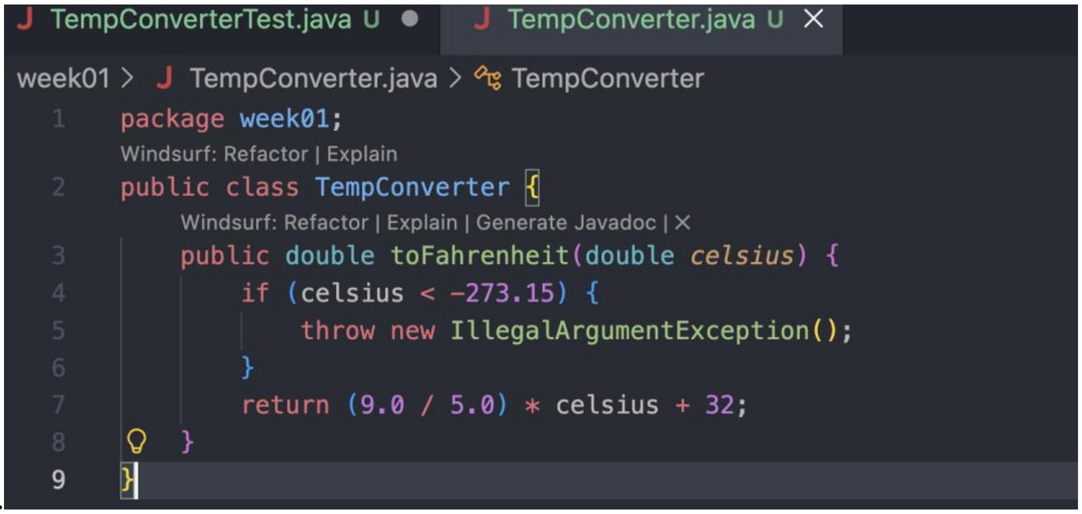


```java
public boolean isCollinear(Point p1, Point p2) { //f
    if (this.x == p1.x && this.x == p2.x) {
        return true;
    }
    if (this.y == p1.y && this.y == p2.y) {
        return true;
    }

    double slope1 = roundSlope(this, p1);
    double slope2 = roundSlope(this, p2);

    return slope1 == slope2;
}

Windsurf: Refactor | Explain | Generate Javadoc | X
private double roundSlope(Point a, Point b) {
    if (a.x == b.x) {
        return Double.POSITIVE_INFINITY; // vertical line/infinitely big slope
    }
    double slope = (double) (b.y - a.y) / (b.x - a.x);
    return Math.round(slope * 10000.0) / 10000.0; // round to 4 decimal places
}
```

8.



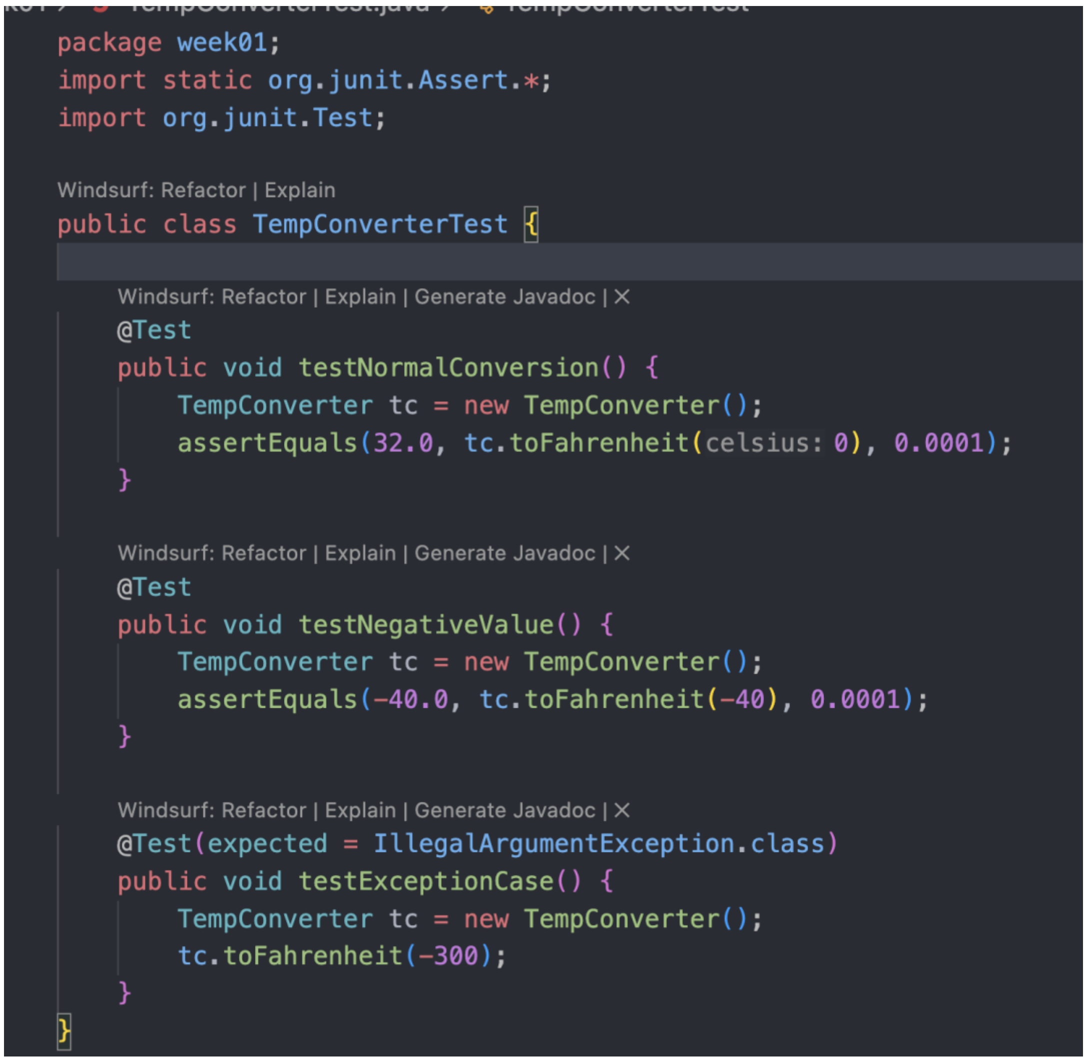
```java
public class Stock {
    private String symbol;
    private int totalShares;
    private double totalCost;

    public Stock(String theSymbol) {
        symbol = theSymbol;
        totalShares = 0;
        totalCost = 0.0;
    }

    Windsurf: Refactor | Explain | Generate Javadoc | X
    public double getProfit(double currentPrice) {
        return totalShares * currentPrice - totalCost;
    }

    Windsurf: Refactor | Explain | Generate Javadoc | X
    public void purchase(int shares, double pricePerShare) {
        totalShares += shares;
        totalCost += shares * pricePerShare;
    }

    Windsurf: Refactor | Explain | Generate Javadoc | X
    public String getSymbol() {
        return symbol;
    }

    Windsurf: Refactor | Explain | Generate Javadoc | X
    public int getTotalShares() {
        return totalShares;
    }

    Windsurf: Refactor | Explain | Generate Javadoc | X
    public double getTotalCost() {
        return totalCost;
    }

    Windsurf: Refactor | Explain | Generate Javadoc | X
    public void clear() { // a
        totalShares = 0;
        totalCost = 0.0;
    }
}
```

9. 
TempConverterTest.java U ● TempConverter.java U ×

week01 > TempConverter.java > TempConverter

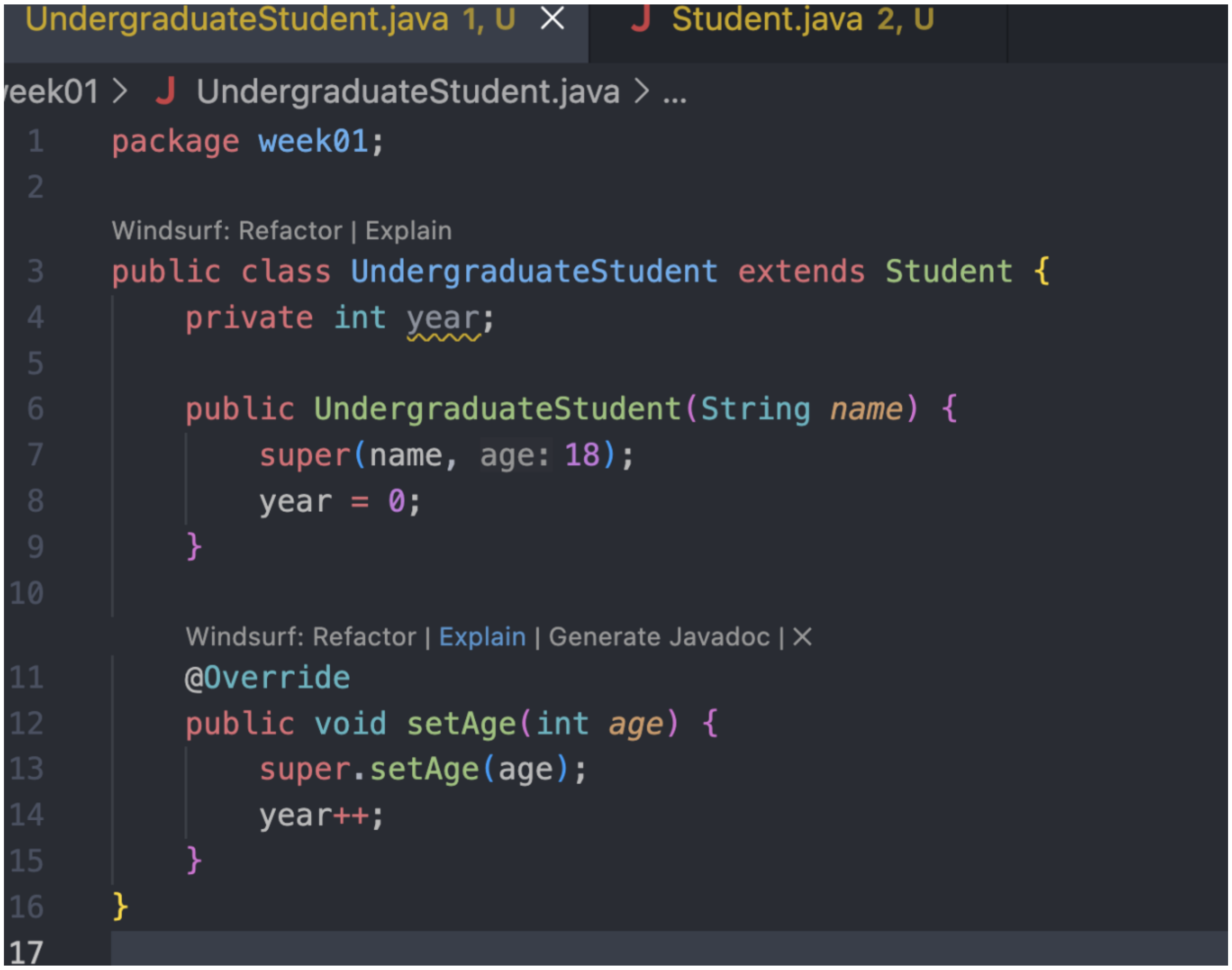
```java
package week01;
public class TempConverter {
    public double toFahrenheit(double celsius) {
        if (celsius < -273.15) {
            throw new IllegalArgumentException();
        }
        return (9.0 / 5.0) * celsius + 32;
    }
}
```

Junit tests:



```java
package week01;
import static org.junit.Assert.*;
import org.junit.Test;

public class TempConverterTest {

    @Test
    public void testNormalConversion() {
        TempConverter tc = new TempConverter();
        assertEquals(32.0, tc.toFahrenheit(celsius: 0), 0.0001);
    }

    @Test
    public void testNegativeValue() {
        TempConverter tc = new TempConverter();
        assertEquals(-40.0, tc.toFahrenheit(-40), 0.0001);
    }

    @Test(expected = IllegalArgumentException.class)
    public void testExceptionCase() {
        TempConverter tc = new TempConverter();
        tc.toFahrenheit(-300);
    }
}
```

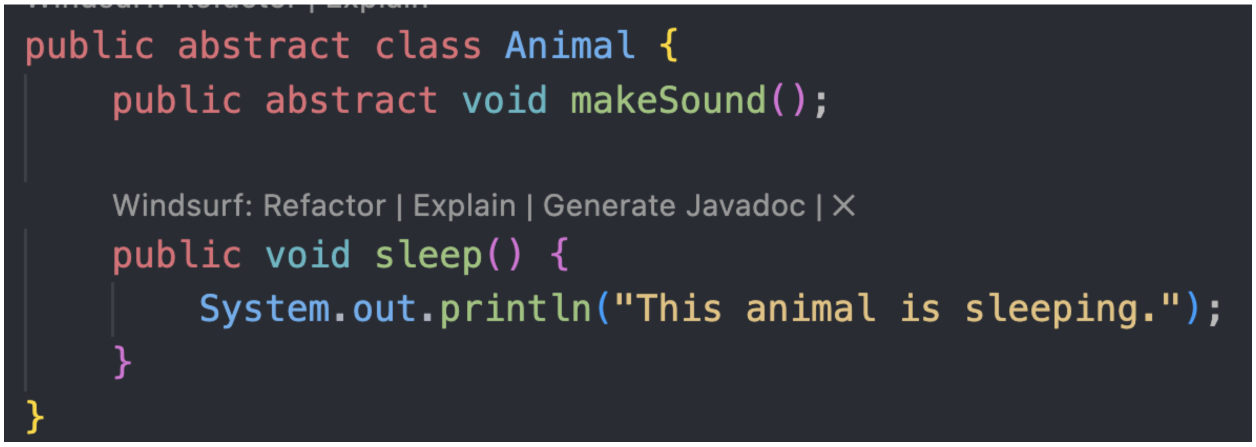
10.


UndergraduateStudent.java 1, U × Student.java 2, U

eek01 > UndergraduateStudent.java > ...

```java
package week01;

Windsurf: Refactor | Explain
public class UndergraduateStudent extends Student {
    private int year;

    public UndergraduateStudent(String name) {
        super(name, age: 18);
        year = 0;
    }

    Windsurf: Refactor | Explain | Generate Javadoc | ×
    @Override
    public void setAge(int age) {
        super.setAge(age);
        year++;
    }
}
```

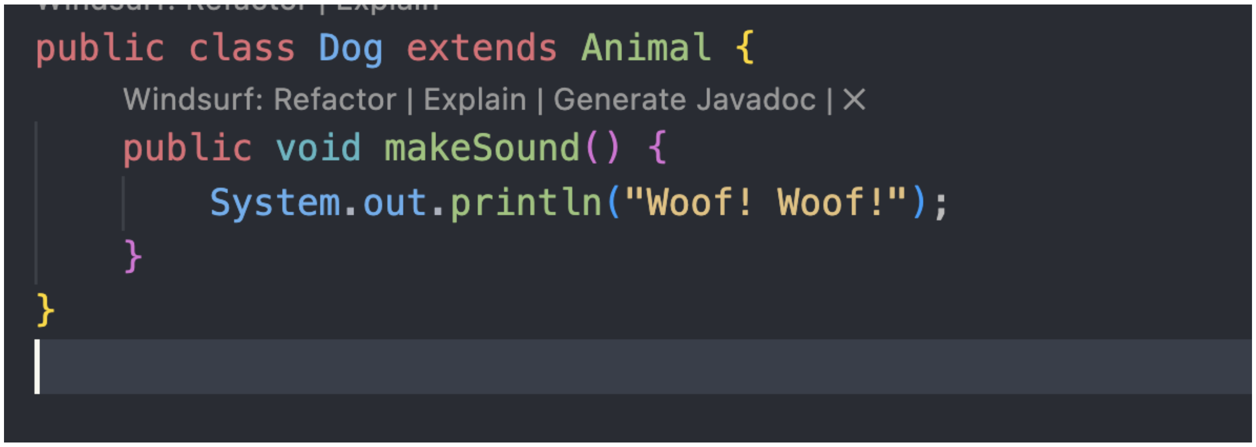
11.



```java
public abstract class Animal {
    public abstract void makeSound();

    Windsurf: Refactor | Explain | Generate Javadoc | ×
    public void sleep() {
        System.out.println("This animal is sleeping.");
    }
}
```

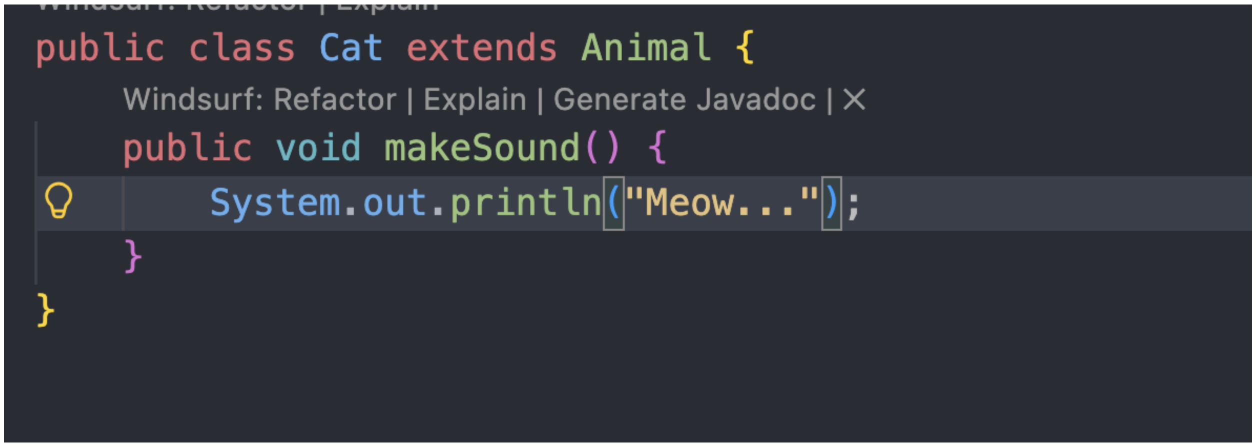
12.

Dog:



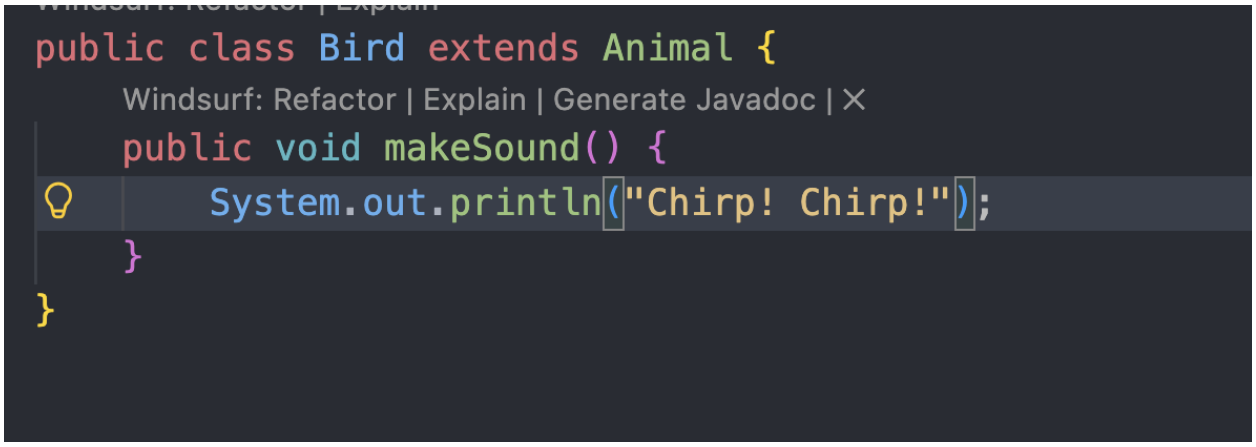
```java
public class Dog extends Animal {
    public void makeSound() {
        System.out.println("Woof! Woof!");
    }
}
```

Cat:



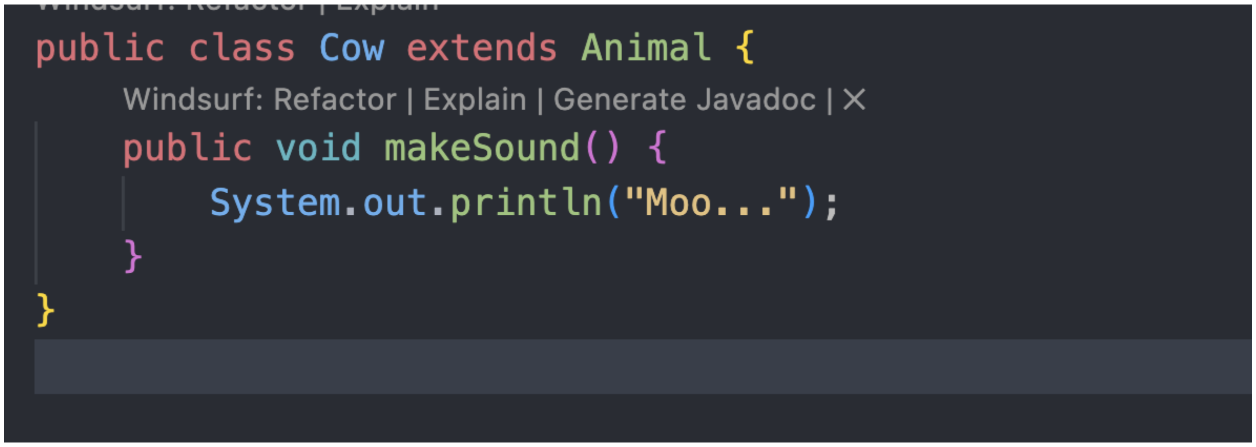
```java
public class Cat extends Animal {
    public void makeSound() {
        System.out.println("Meow...");
    }
}
```

Bird:



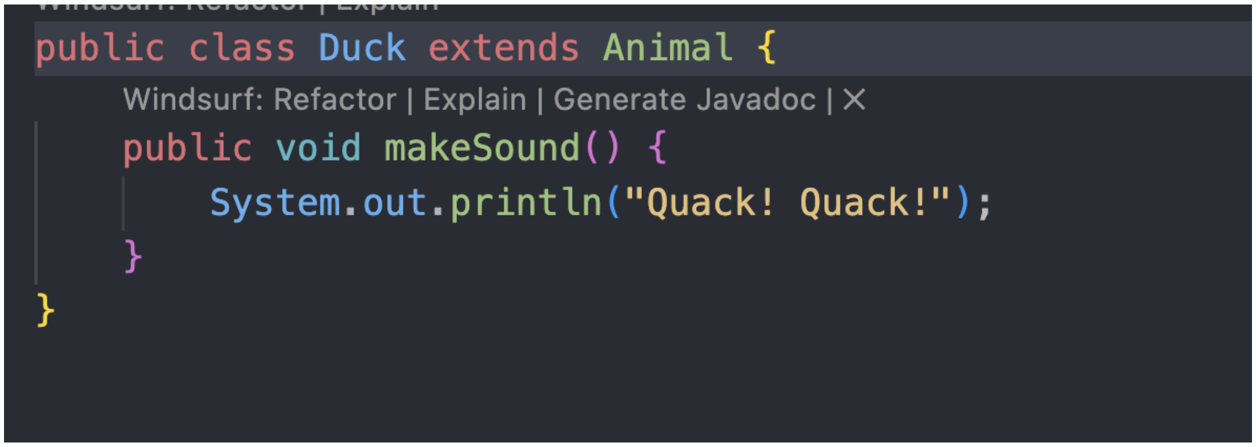
```java
public class Bird extends Animal {
    public void makeSound() {
        System.out.println("Chirp! Chirp!");
    }
}
```

Cow:



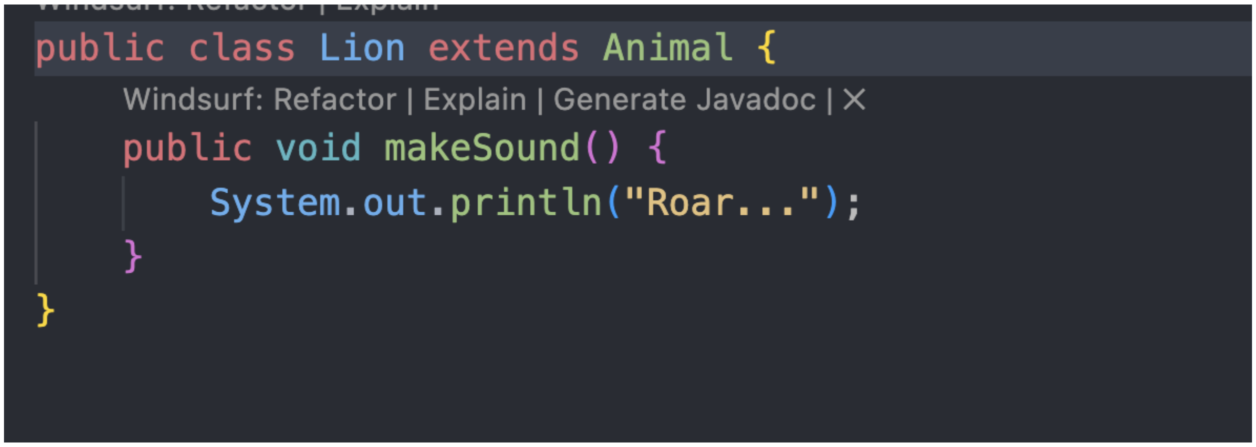
```java
public class Cow extends Animal {
    public void makeSound() {
        System.out.println("Moo...");
    }
}
```

Duck:



```java
public class Duck extends Animal {
    public void makeSound() {
        System.out.println("Quack! Quack!");
    }
}
```

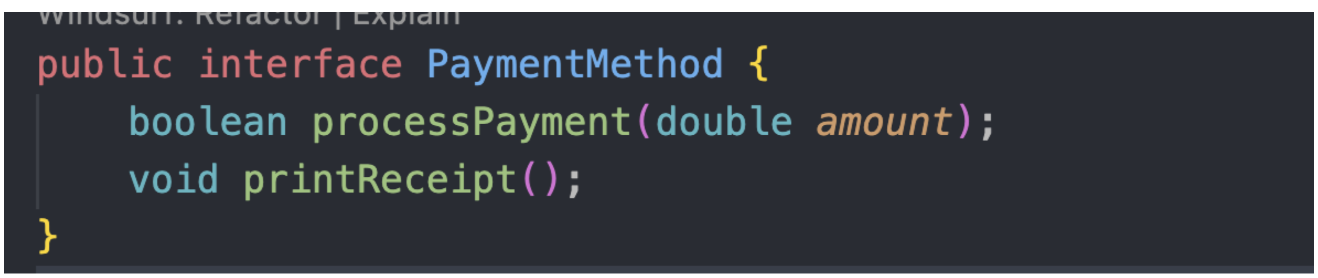
Lion:



```java
public class Lion extends Animal {
    public void makeSound() {
        System.out.println("Roar...");
    }
}
```

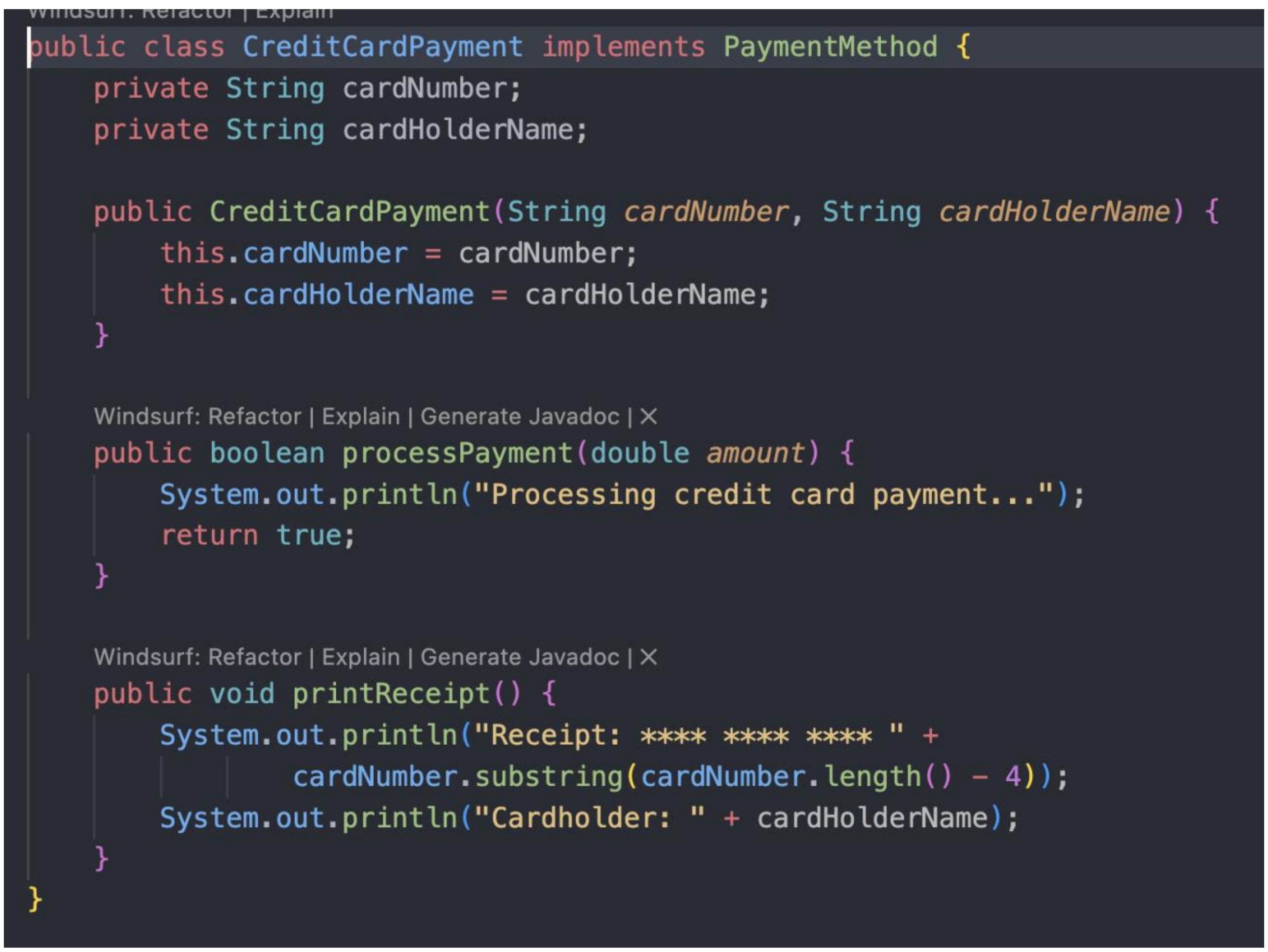
13.

PaymentMethod interface:



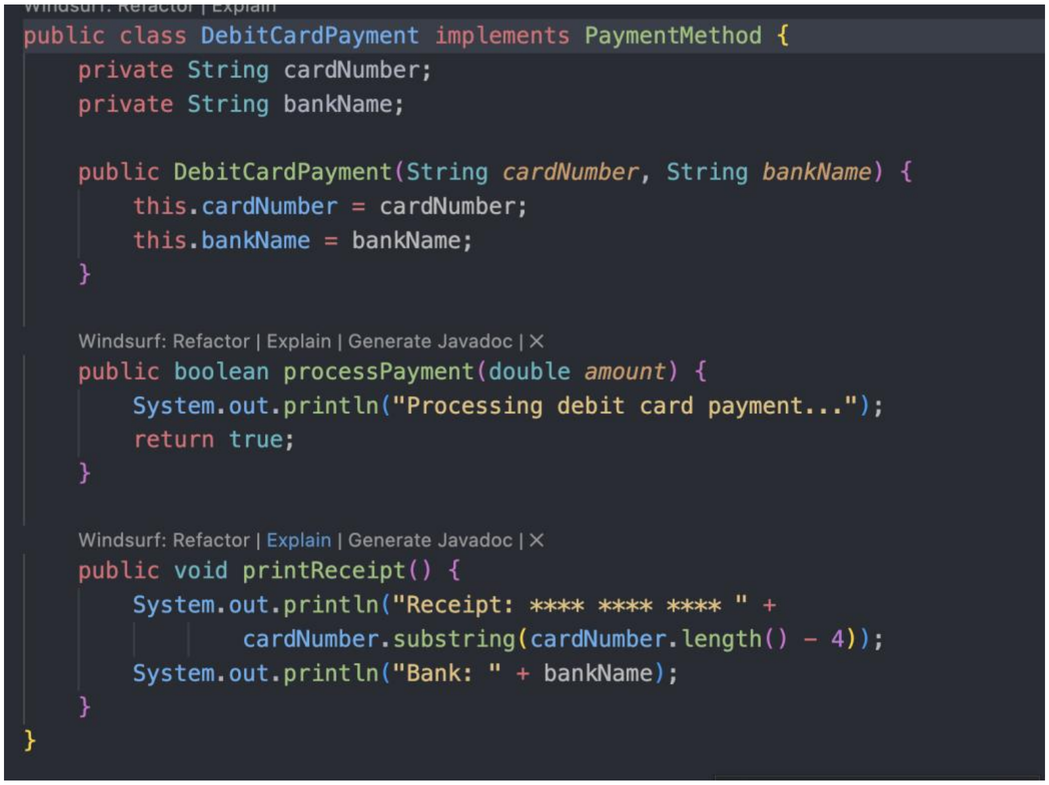
```java
public interface PaymentMethod {
    boolean processPayment(double amount);
    void printReceipt();
}
```

CreditCardPayment:



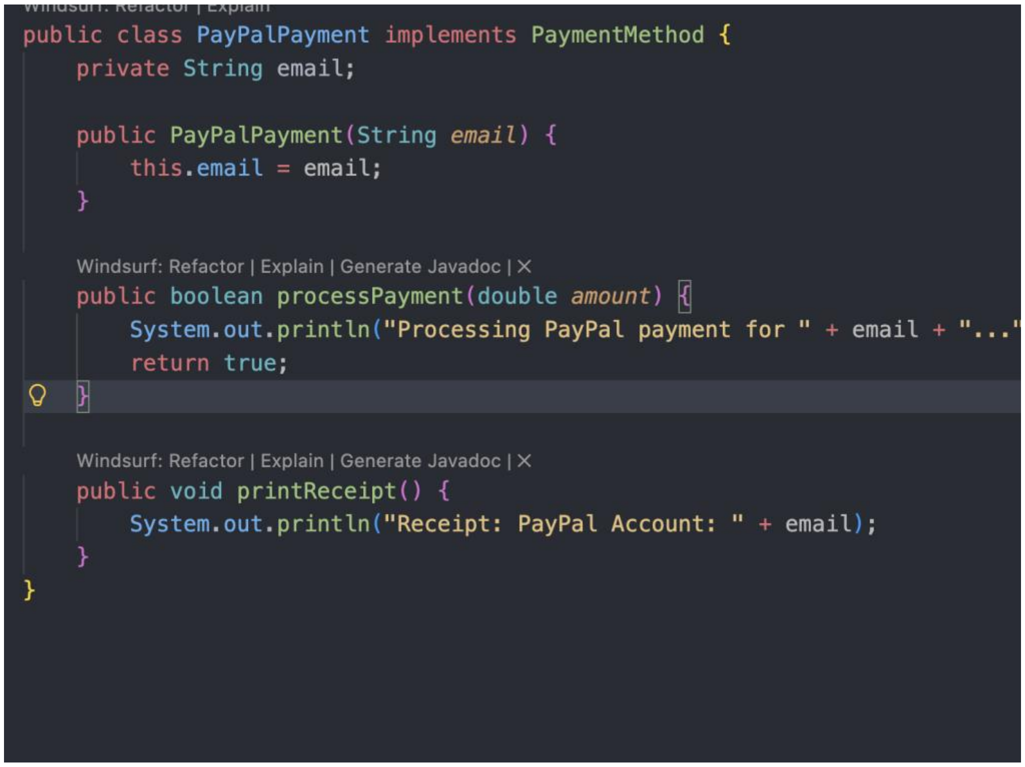
```java
public class CreditCardPayment implements PaymentMethod {
    private String cardNumber;
    private String cardHolderName;

    public CreditCardPayment(String cardNumber, String cardHolderName) {
        this.cardNumber = cardNumber;
        this.cardHolderName = cardHolderName;
    }

    Windsurf: Refactor | Explain | Generate Javadoc | ×
    public boolean processPayment(double amount) {
        System.out.println("Processing credit card payment...");
        return true;
    }

    Windsurf: Refactor | Explain | Generate Javadoc | ×
    public void printReceipt() {
        System.out.println("Receipt: **** **** **** " +
                cardNumber.substring(cardNumber.length() - 4));
        System.out.println("Cardholder: " + cardHolderName);
    }
}
```

DebitCardPayment:

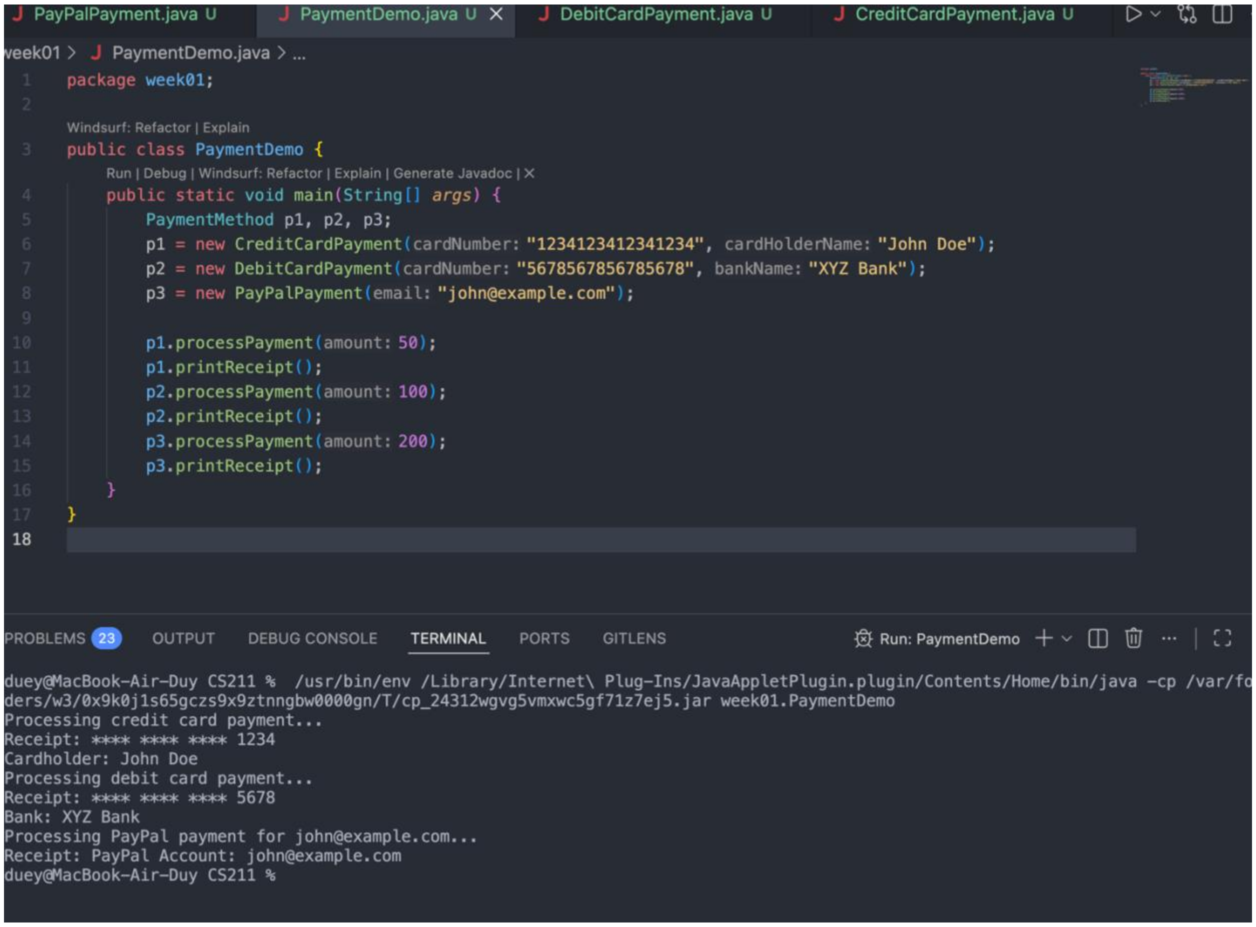


```
Windsurf: Refactor | Explain
public class DebitCardPayment implements PaymentMethod {
    private String cardNumber;
    private String bankName;

    public DebitCardPayment(String cardNumber, String bankName) {
        this.cardNumber = cardNumber;
        this.bankName = bankName;
    }

    Windsurf: Refactor | Explain | Generate Javadoc | ×
    public boolean processPayment(double amount) {
        System.out.println("Processing debit card payment...");
        return true;
    }

    Windsurf: Refactor | Explain | Generate Javadoc | ×
    public void printReceipt() {
        System.out.println("Receipt: **** **** **** " +
                cardNumber.substring(cardNumber.length() - 4));
        System.out.println("Bank: " + bankName);
    }
}
```

PayPalPayment:



```
Windsurf: Refactor | Explain
public class PayPalPayment implements PaymentMethod {
    private String email;

    public PayPalPayment(String email) {
        this.email = email;
    }

    Windsurf: Refactor | Explain | Generate Javadoc | ×
    public boolean processPayment(double amount) {
        System.out.println("Processing PayPal payment for " + email + "..."
        return true;
    }

    Windsurf: Refactor | Explain | Generate Javadoc | ×
    public void printReceipt() {
        System.out.println("Receipt: PayPal Account: " + email);
    }
}
```



```java
package week01;

public class PaymentDemo {
    public static void main(String[] args) {
        PaymentMethod p1, p2, p3;
        p1 = new CreditCardPayment(cardNumber: "1234123412341234", cardHolderName: "John Doe");
        p2 = new DebitCardPayment(cardNumber: "5678567856785678", bankName: "XYZ Bank");
        p3 = new PayPalPayment(email: "john@example.com");

        p1.processPayment(amount: 50);
        p1.printReceipt();
        p2.processPayment(amount: 100);
        p2.printReceipt();
        p3.processPayment(amount: 200);
        p3.printReceipt();
    }
}
```

```
duey@MacBook-Air-Duy CS211 %  /usr/bin/env /Library/Internet\ Plug-Ins/JavaAppletPlugin.plugin/Contents/Home/bin/java -cp /var/fo
ders/w3/0x9k0j1s65gczs9x9ztnngbw0000gn/T/cp_24312wgvg5vmxwc5gf71z7ej5.jar week01.PaymentDemo
Processing credit card payment...
Receipt: **** **** **** 1234
Cardholder: John Doe
Processing debit card payment...
Receipt: **** **** **** 5678
Bank: XYZ Bank
Processing PayPal payment for john@example.com...
Receipt: PayPal Account: john@example.com
duey@MacBook-Air-Duy CS211 %
```

14.

[1, 2, 6, 8]

[10, 30, 40, 20, 60, 50]

[-4, 1, 25, 4, 16, 9, 64, 36, 49]

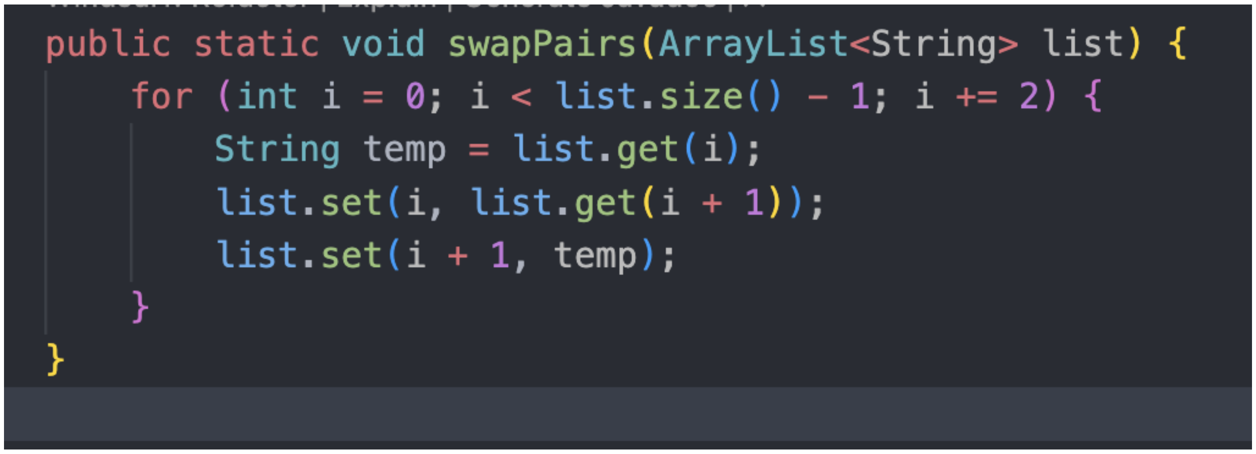
15.

[31, 21, 11]

[5, 8, 10, 3, 9]

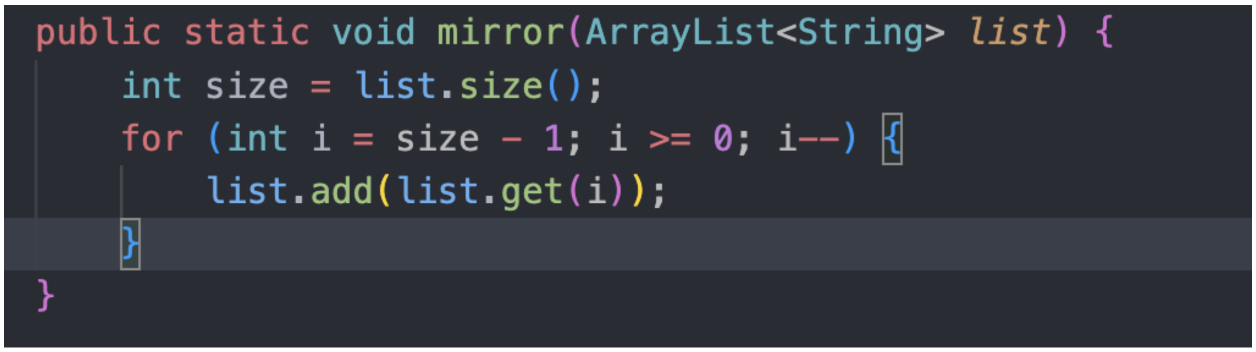
[34, 10, 18, 29, 4, 0]

16.



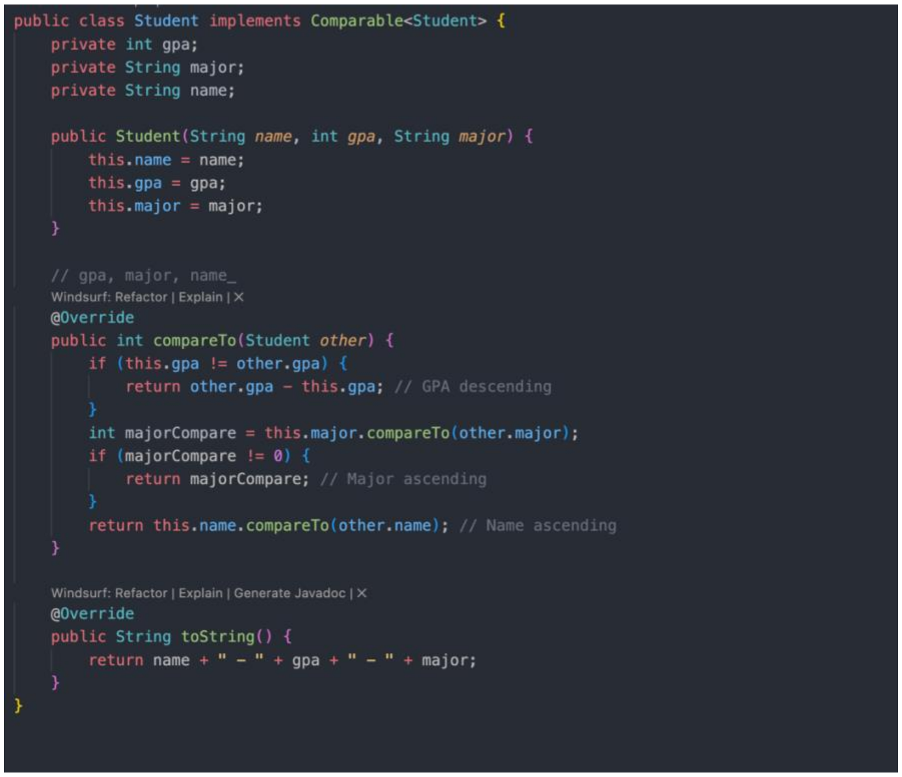
```java
public static void swapPairs(ArrayList<String> list) {
    for (int i = 0; i < list.size() - 1; i += 2) {
        String temp = list.get(i);
        list.set(i, list.get(i + 1));
        list.set(i + 1, temp);
    }
}
```

17.

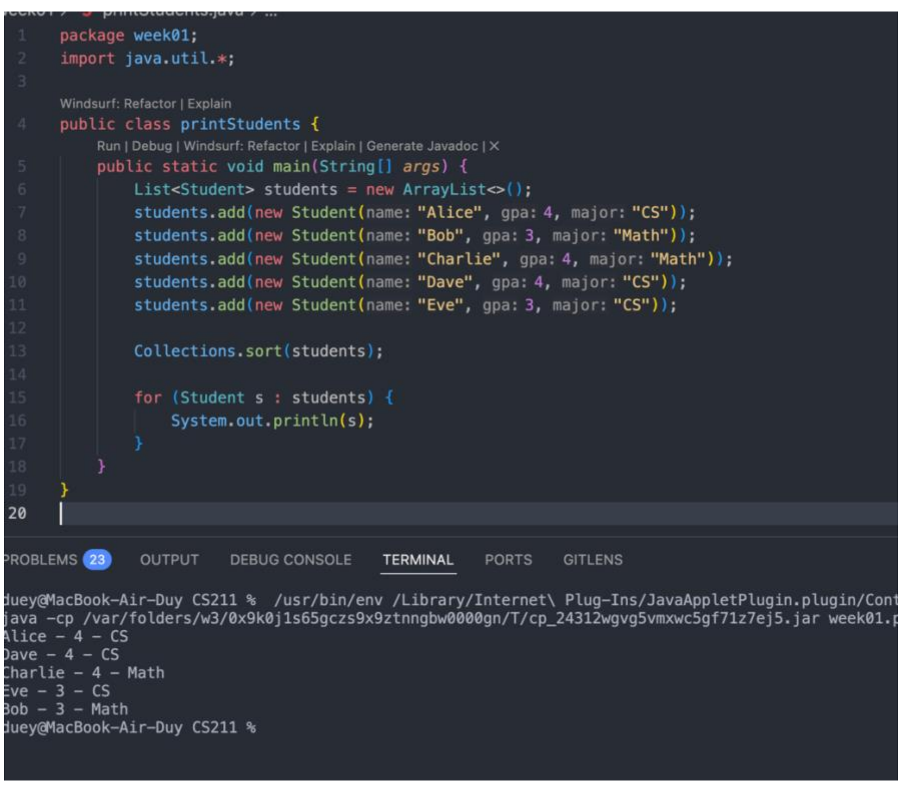


```java
public static void mirror(ArrayList<String> list) {
    int size = list.size();
    for (int i = size - 1; i >= 0; i--) {
        list.add(list.get(i));
    }
}
```

18.


Windsurf: Refactor | Explain | ✕
Windsurf: Refactor | Explain | Generate Javadoc | ✕

```java
public class Student implements Comparable<Student> {
    private int gpa;
    private String major;
    private String name;

    public Student(String name, int gpa, String major) {
        this.name = name;
        this.gpa = gpa;
        this.major = major;
    }

    // gpa, major, name_
    @Override
    public int compareTo(Student other) {
        if (this.gpa != other.gpa) {
            return other.gpa - this.gpa; // GPA descending
        }
        int majorCompare = this.major.compareTo(other.major);
        if (majorCompare != 0) {
            return majorCompare; // Major ascending
        }
        return this.name.compareTo(other.name); // Name ascending
    }

    @Override
    public String toString() {
        return name + " - " + gpa + " - " + major;
    }
}
```


Windsurf: Refactor | Explain
Run | Debug | Windsurf: Refactor | Explain | Generate Javadoc | ✕
PROBLEMS 23
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
GITLENS

```java
package week01;
import java.util.*;

public class printStudents {
    public static void main(String[] args) {
        List<Student> students = new ArrayList<>();
        students.add(new Student(name: "Alice", gpa: 4, major: "CS"));
        students.add(new Student(name: "Bob", gpa: 3, major: "Math"));
        students.add(new Student(name: "Charlie", gpa: 4, major: "Math"));
        students.add(new Student(name: "Dave", gpa: 4, major: "CS"));
        students.add(new Student(name: "Eve", gpa: 3, major: "CS"));

        Collections.sort(students);

        for (Student s : students) {
            System.out.println(s);
        }
    }
}
```

```
duey@MacBook-Air-Duy CS211 %  /usr/bin/env /Library/Internet\ Plug-Ins/JavaAppletPlugin.plugin/Cont
java -cp /var/folders/w3/0x9k0j1s65gczs9x9ztnngbw0000gn/T/cp_24312wgvg5vmxwc5gf71z7ej5.jar week01.p
Alice - 4 - CS
Dave - 4 - CS
Charlie - 4 - Math
Eve - 3 - CS
Bob - 3 - Math
duey@MacBook-Air-Duy CS211 %
```

19.

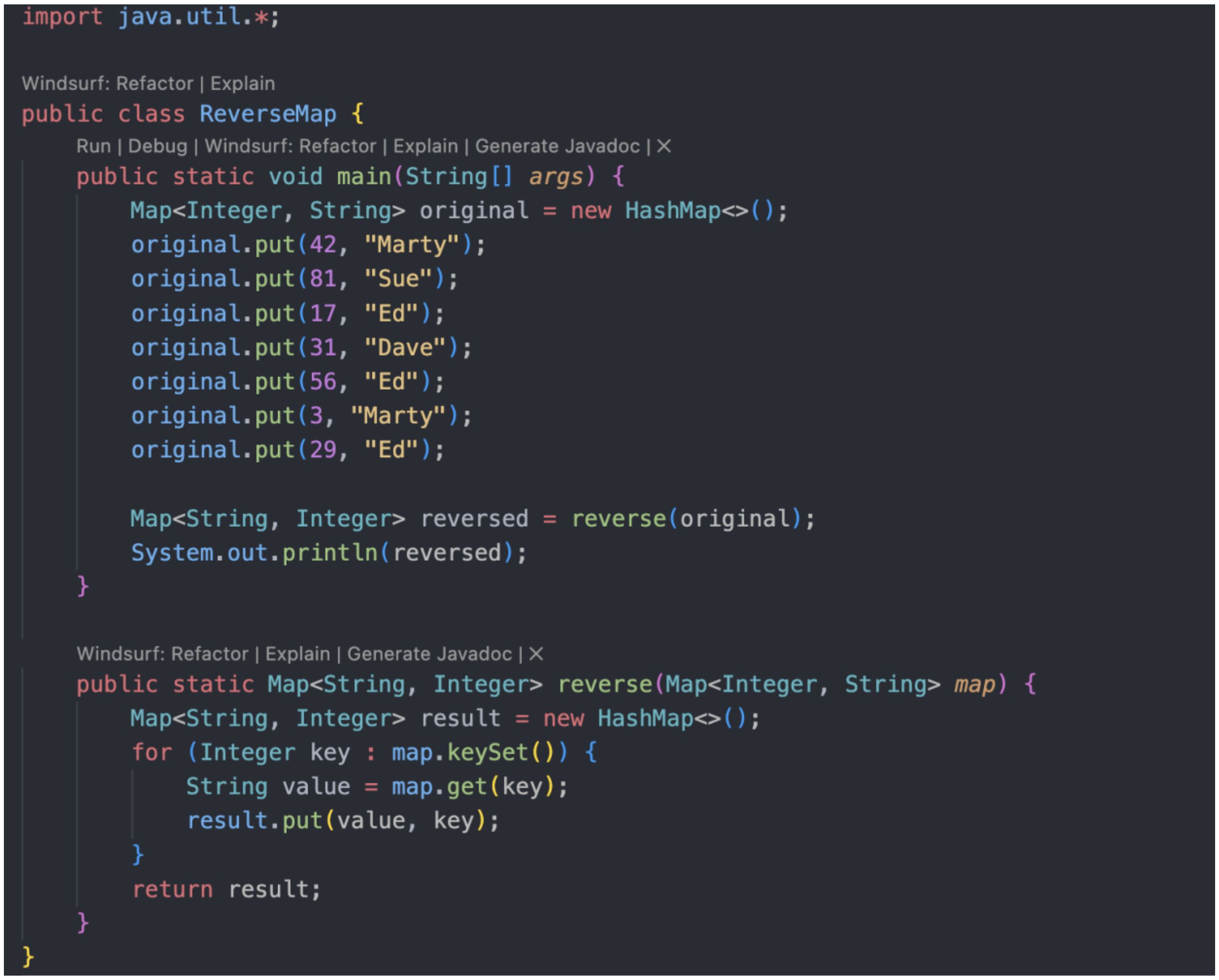
{cing=five, deux=two, four=quatre, one=un, three=trois}

{board=skate, car=drive, computer=play}

{begin=end, boy=girl, ebert=siskel, first=last, heads=tails}

{cotton=rain, light=tree, seed=tree, tree=violin}

20.



```java
import java.util.*;

Windsurf: Refactor | Explain
public class ReverseMap {
    Run | Debug | Windsurf: Refactor | Explain | Generate Javadoc | X
    public static void main(String[] args) {
        Map<Integer, String> original = new HashMap<>();
        original.put(42, "Marty");
        original.put(81, "Sue");
        original.put(17, "Ed");
        original.put(31, "Dave");
        original.put(56, "Ed");
        original.put(3, "Marty");
        original.put(29, "Ed");

        Map<String, Integer> reversed = reverse(original);
        System.out.println(reversed);
    }

    Windsurf: Refactor | Explain | Generate Javadoc | X
    public static Map<String, Integer> reverse(Map<Integer, String> map) {
        Map<String, Integer> result = new HashMap<>();
        for (Integer key : map.keySet()) {
            String value = map.get(key);
            result.put(value, key);
        }
        return result;
    }
}
```