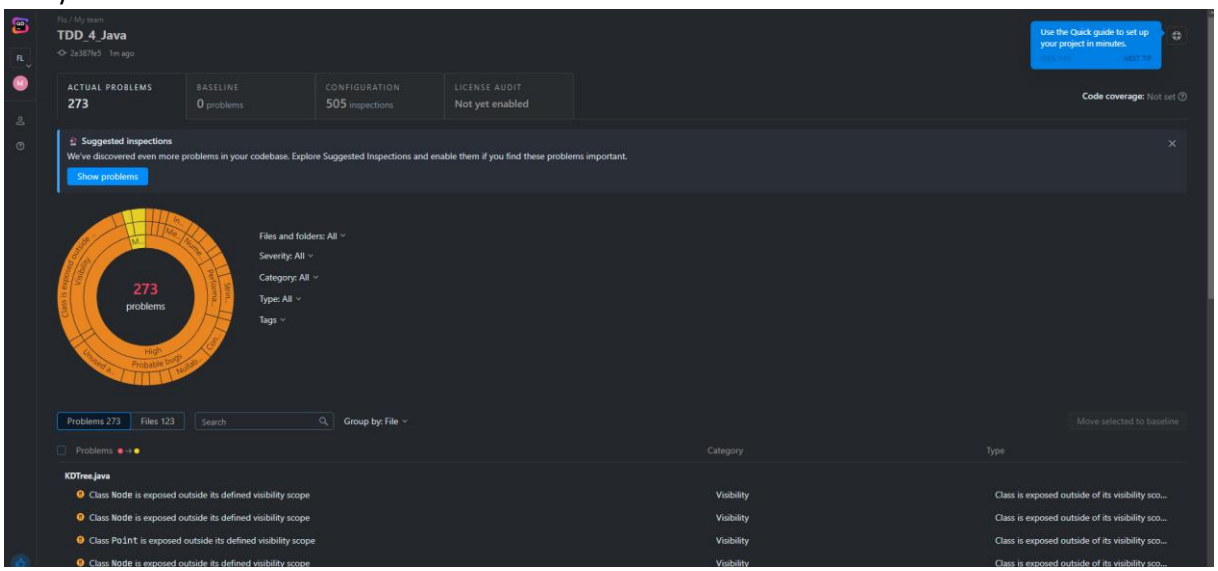
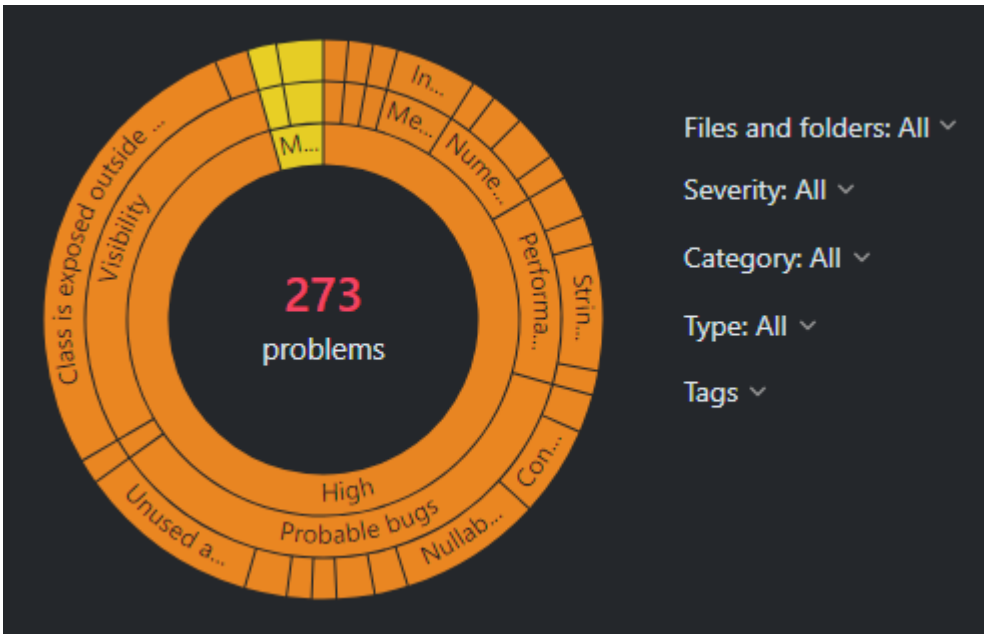
TDD 4

Statische Code Analyse

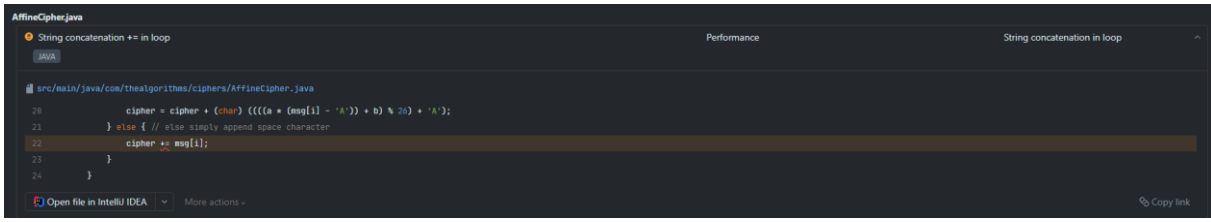
Mit Quodana und IntelliJ anhand des Github Repositorys: https://github.com/TheAlgorithms/Java

1. Anlayse laufen lassen




2. Verbessern
    a. String concatenation += in loop



```java
static String encryptMessage(char[] msg) {
    /// Cipher Text initially empty
    String cipher = "";
    for (int i = 0; i < msg.length; i++) {
        // Avoid space to be encrypted
        /* applying encryption formula ( a x + b ) mod m
        {here x is msg[i] and m is 26} and added 'A' to
        bring it in range of ascii alphabet[ 65-90 | A-Z ] */
        if (msg[i] != ' ') {
            cipher = cipher + (char) ((((a * (msg[i] - 'A')) + b) % 26) + 'A');
        } else { // else simply append space character
            cipher += msg[i];
        }
    }
    return cipher;
}
```

String concatenation '+=' in loop

Convert variable 'cipher' from String to StringBuilder Alt+Umschalt+Eingabe More ac

```java
static String encryptMessage(char[] msg) {
    /// Cipher Text initially empty
    StringBuilder cipher = new StringBuilder();
    for (int i = 0; i < msg.length; i++) {
        // Avoid space to be encrypted
        /* applying encryption formula ( a x + b ) mod m
        {here x is msg[i] and m is 26} and added 'A' to
        bring it in range of ascii alphabet[ 65-90 | A-Z ] */
        if (msg[i] != ' ') {
            cipher.append((char) ((((a * (msg[i] - 'A')) + b) % 26) + 'A'));
        } else { // else simply append space character
            cipher.append(msg[i]);
        }
    }
    return cipher.toString();
}
```

b. Manual array copy

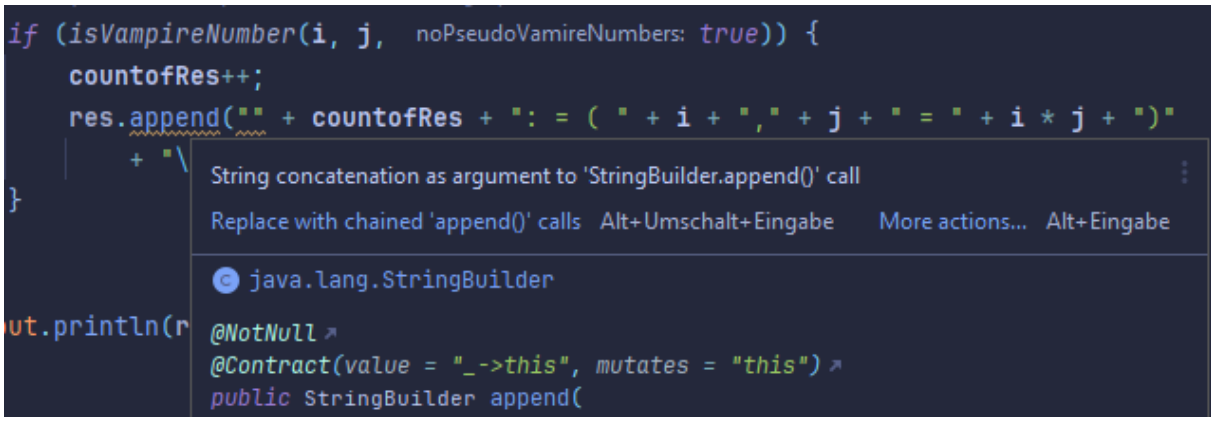
```java
public void floydwarshall(int[][] adjacencyMatrix) { // calculates all the distances from
    for (int source = 1; source <= numberofvertices; source++) {
        for (int destination = 1; destination <= numberofvertices; destination++) {
                                                                    estination];
        }
    }
    for (int intermediate = 1; intermediate <= numberofvertices; intermediate++) {
```

Manual array copy

Replace with 'System.arraycopy()' Alt+Umschalt+Eingabe More actions... Alt+Eingabe

```
for (int source = 1; source <= numberofvertices; source++) {
    System.arraycopy(adjacencyMatrix[source], srcPos: 1, distanceMatrix[source], destPos: 1, numberofvertices);
}
```

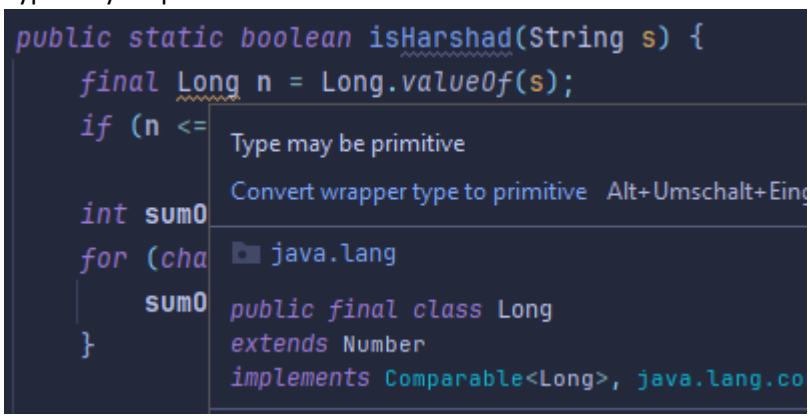
c. String concatenation as argument to StringBuilder.append() call




```
static void test(int startValue, int stopValue) {
    int countofRes = 1;
    StringBuilder res = new StringBuilder();

    for (int i = startValue; i <= stopValue; i++) {
        for (int j = i; j <= stopValue; j++) {
            // System.out.println(i+ " "+ j);
            if (isVampireNumber(i, j, noPseudoVamireNumbers: true)) {
                countofRes++;
                String countString = String.valueOf(countofRes);
                String iString = String.valueOf(i);
                String jString = String.valueOf(j);
                String productString = String.valueOf( i * j);
                res.append(countString).append(": = ( ").append(iString).append(",").append(jString).append(" = ").append(productString).append(")\n");
            }
        }
    }
    System.out.println(res);
}
```

d. Type may be primitive



```java
public static boolean isHarshad(String s)
    final long n = Long.parseLong(s);
    if (n <= 0) return false;

    int sumOfDigits = 0;
    for (char ch : s.toCharArray()) {
        sumOfDigits += ch - '0';
    }

    return n % sumOfDigits == 0;
```

e. Condition isMaximizer is always true

```java
public static void main(String[] args) {
    MiniMaxAlgorithm miniMaxAlgorith = new MiniMaxAlgorithm();
    boolean isMaximizer = true; // Specifies the player that goes first.
    boolean verbose = true; // True to show each players choices.
    int bestScore;

    bestScore = miniMaxAlgorith.miniMax( depth: 0, isMaximizer,  index: 0, verbose);

    if (verbose) {
        System.out.println();
    }

    System.out.println(Arrays.toString(miniMaxAlgorith.getScores()));
    System.out.println("The best score for " + (isMaximizer ? "Maximizer" : "Minimizer") + " is " + bestScore);
}
```

Condition 'isMaximizer' is always 'true'

Simplify 'isMaximizer' to true Alt+Umschalt+Eingabe More actions... Alt+Eingabe

```java
public static void main(String[] args) {
    MiniMaxAlgorithm miniMaxAlgorith = new MiniMaxAlgorithm();
    boolean isMaximizer = true; // Specifies the player that goes first.
    boolean verbose = true; // True to show each players choices.
    int bestScore;

    bestScore = miniMaxAlgorith.miniMax( depth: 0, isMaximizer,  index: 0, verbose);

    System.out.println();

    System.out.println(Arrays.toString(miniMaxAlgorith.getScores()));
    System.out.println("The best score for " + "Maximizer" + " is " + bestScore);
}
```

f. Condition x > 1.00000 is always false when reached

```
public void changeMess() {
    for (int y : message) {
        double x = randomGenerator.nextDouble();
        while (x < 0.0000 || x > 1.00000) {
            x = randomGener
        }
        if (x < ber) {
            messageChanged = true;
            if (y == 1) {
                message.set(message.indexOf(y), 0);
            } else {
                message.set(message.indexOf(y), 1);
            }
        }
    }
```

Condition 'x < 0.0000 || x > 1.00000' is always 'false'

Remove 'while' statement Alt+Umschalt+Eingabe Mor

```
public void changeMess() {
    for (int y : message) {
        double x = randomGenerator.nextDouble();
        if (x < ber) {
            messageChanged = true;
            if (y == 1) {
                message.set(message.indexOf(y), 0);
            } else {
                message.set(message.indexOf(y), 1);
            }
        }
    }
```

g. Number objects are compared using ==, not 'equals()'

```
if (x.size() == p.size()) {
    for (int i = 0; i < p.size(); i++) {
        if (x.get(i) == p.get(i)) {
            x.set(i, 0
        } else {
            x.set(i, 1
        }
    }
    for (int i = 0; i < x.size() && x.get(i) != 1; i++) {
        x.remove( index: 0);
    }
```

Number objects are compared using '==', not 'equals()'

Replace '==' with null-safe 'equals()' Alt+Umschalt+Eingabe More actions... Alt+Eingabe

```java
if (x.size() == p.size()) {
    for (int i = 0; i < p.size(); i++) {
        if (Objects.equals(x.get(i), p.get(i))) {
            x.set(i, 0);
        } else {
            x.set(i, 1);
        }
    }
```

h. while statement has empty body

```java
@Override
public <T extends Comparable<T>> T[] sort(T[] array) {
    int n = array.length;
    if (n == 0) {
        return array;
    }
    while (doSort(array, left: 0, right: n - 1)) {
    }
    re
}
```

'while' statement has empty body

```java
@Override
public <T extends Comparable<T>> T[] sort(T[] array) {
    int n = array.length;
    if (n == 0) {
        return array;
    }

    boolean swapped;
    do {
        // In jeder Iteration wird das Array erneut sortiert
        swapped = doSort(array, left: 0, right: n - 1);
    } while (swapped);

    return array;
}
```

i. Variable j initializer 0 is redundant

```java
private static void searchPat(String text, String pattern, int q) {
    int m = pattern.length();
    int n = text.length();
    int t = 0;
    int p = 0;
    int h = 1;
    int j = 0;
    int i = 0;
```

```java
private static void searchPat(String text, String pattern, int q) {
    int m = pattern.length();
    int n = text.length();
    int t = 0;
    int p = 0;
    int h;
    int j;
    int i;
```

j. Empty catch block

```java
try {
    t.join();
    t1.join();
    t2.join();
    t3.join();
} catch (InterruptedException e) {
}
boolean found = t.getResult() || t1.getResul
```

```java
try {
    t.join();
    t1.join();
    t2.join();
    t3.join();
} catch (InterruptedException e) {
    Thread.currentThread().interrupt(); // Wiederherstellen des Unterbrechungsstatus
    System.err.println("Interrupted while waiting for threads to finish.");
}
```