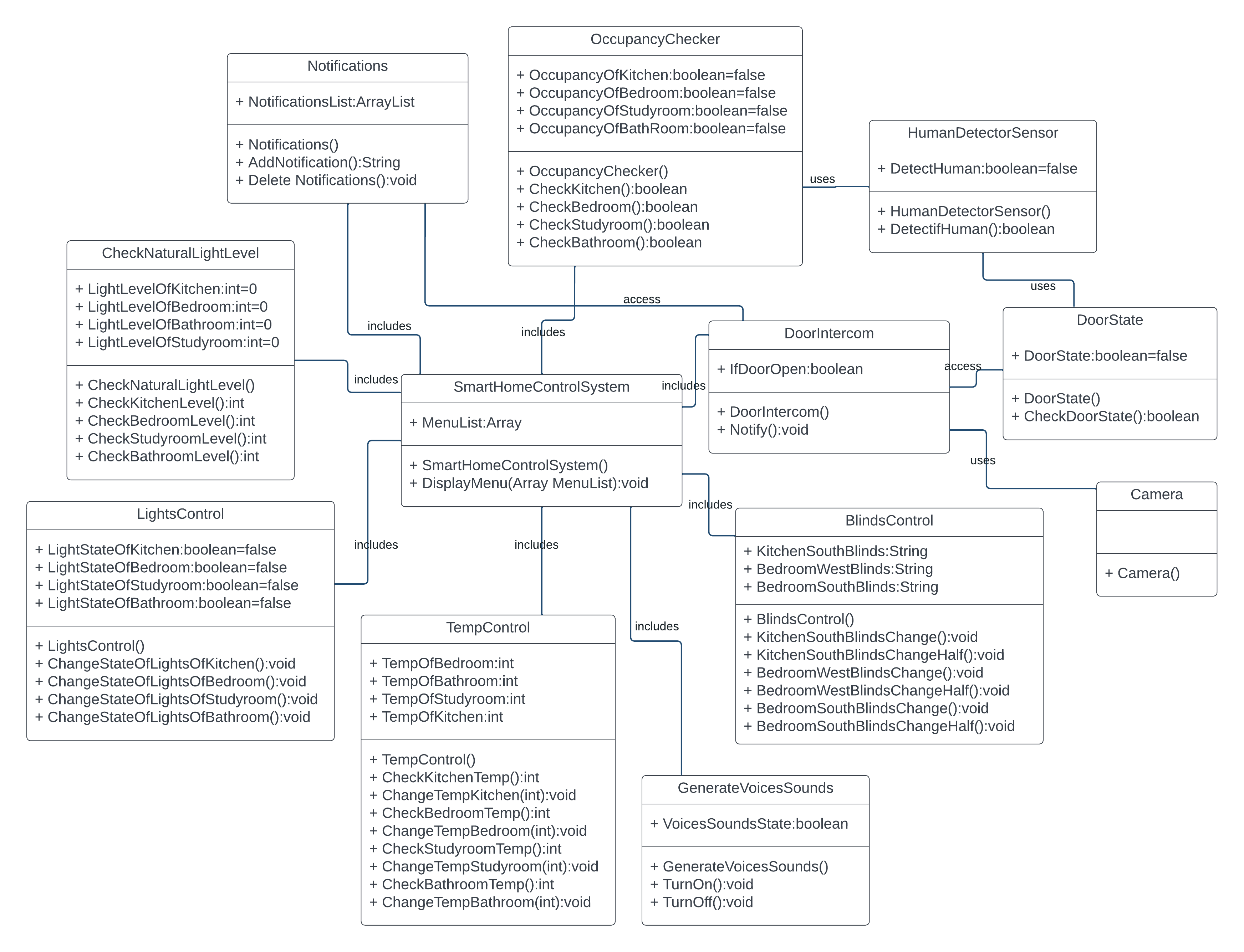
# Question 1:

Question 2:

# IMG-20231229-WA0008.jpg

# Question 3:

# a)project%203%20(2).jpg

# b)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TestCase Number** | **Unit Under Test** | **a** | **b** | **c** | **d** | **Expected output** |
| TC#1 | Numbered 2 | 0 | 1 | 2 | 3 | InvalidSideException |
| TC#2 | Numbered 3 | 4 | 4 | 4 | 4 | Square |
| TC#3 | Numbered 6 | 6 | 8 | 6 | 8 | Rectangle |
| TC#4 | Numbered 7 | 4 | 5 | 6 | 4 | Trapezoid |
| TC#5 | Numbered 1 | 4 | 5 | 6 | 7 | Trapezoid |
| TC#6 | Numbered 4 | q | 3 | 4 | 7 | InvalidSideException |
| TC#7 | Numbered 5 | -2 | 1 | 4 | 3 | InvalidSideException |
| TC#8 | Numbered 8 | 9 | 8 | 9 | - | NotAValidTetrago |
| TC#9 | Numbered 2 | 3 | 0 | 2 | 0 | InvalidSideException |
| TC#10 | Numbered 8 | 11 | 8 | - | - | NotAValidTetrago |
| TC#11 | Numbered 3 | 12 | 12 | 12 | 12 | Square |
| TC#12 | Numbered 6 | 5 | 13 | 5 | 13 | Rectangle |

# Question 4:

# a) Type of Testing:

The test cases listed in the scenarios above represent Unit Testing as they individually test the identifyTetragon method and are aimed at verifying its functionalities.

# b) Integration Test Scenario Explanation:

Suppose test cases for identifyTetragon and printShapeInfo pass individually but fail when integrated. Consider the following situation:

## **Integration Test Scenario:**

A successful identification of a tetragon (let's say it's recognized as a square) by identifyTetragon method is passed as input to printShapeInfo.

The printShapeInfo method tries to print the corresponding Unicode symbol for the recognized shape (in this case, a square).

## **Possible Causes of Integration Test Failure**:

## **1. \*\*Misinterpretation of Output:**

- Scenario: Even if `identifyTetragon` correctly identifies a square, the subsequent handling in `printShapeInfo` might have an unexpected interpretation of the recognized shape.

## -Example:

- While `identifyTetragon` might return "Square" for a set of input sides, `printShapeInfo` might interpret this as a "Circle" due to a logical error in its conditional logic.

- This discrepancy leads to an inconsistency in the output, causing the integration test to fail despite the individual methods functioning correctly.

## **2.Inconsistent Symbol Mapping:**

- Scenario: The mapping of recognized shapes to Unicode symbols in `printShapeInfo` might not align correctly with the actual identified shapes.

## -Example:

## - Suppose `identifyTetragon` correctly identifies a square. However, in `printShapeInfo`, there might be a mistake where the Unicode symbol for a circle is used instead of the symbol for a square.

- This mismatch in symbol mapping leads to an incorrect output symbol despite the correct identification by `identifyTetragon`, resulting in a failed integration test.

## **3.Data Transfer or Input Mismatch:**

-Scenario: The transfer of data between the two methods might encounter an issue, leading to an incorrect input received by `printShapeInfo`.

## -Example:

- If the output from `identifyTetragon` is not passed correctly or if it encounters a formatting issue while being sent to `printShapeInfo`, the latter might receive an unexpected or incorrect input.

- This discrepancy in input causes `printShapeInfo` to process the data inaccurately, leading to a failure in symbol representation despite both methods functioning correctly on their own.

These causes emphasize potential discrepancies in how data is handled, interpreted, or mapped between the methods during their integration, leading to an inconsistency in the expected output despite both individual methods working perfectly in isolation.

## **4.Unexpected Output Handling:**

-Scenario: The integration might fail if printShapeInfo has an unexpected handling or interpretation of the recognized shapes.

-Example: For instance, even though identifyTetragon recognizes a shape as a square, printShapeInfo might misinterpret it due to a logical error in its if-else conditions.

## **5.Unicode Symbol Mapping Issue:**

-Scenario: Despite the correct identification by identifyTetragon, if printShapeInfo has incorrect Unicode mappings for symbols (e.g., a mismatch between the recognized shape and the symbol to be printed).

## Example:

## Suppose identifyTetragon correctly identifies a square, but due to an oversight, printShapeInfo uses the Unicode symbol for a circle instead of a square. This discrepancy in symbol mapping causes the integration test to fail, even though both methods work perfectly in isolation.

This discrepancy could result from an inconsistency in how shapes are recognized and mapped to their respective symbols in the integrated scenario, leading to a failure in the overall integration test despite both individual methods functioning correctly.

## **#Unit Test Scenario for `printShapeInfo` Method:**

Let's create test cases to ensure that `printShapeInfo` method behaves correctly for various shapes by checking the printed Unicode symbols.

## Test Cases:

|  |  |  |  |
| --- | --- | --- | --- |
| Taste Case Number | Unit Under Test | Shape | Expected output |
| TC#1 | printShapeInfo | circle | ❍ |
| TC#2 | printShapeInfo | triangle | △ |
| TC#3 | printShapeInfo | square | □ |
| TC#4 | printShapeInfo | trapezoid | ♢ |
| TC#5 | printShapeInfo | hexagon | "I don't know this shape!!!" |

## #Explanation:

These test cases aim to individually check the behavior of the `printShapeInfo` method for different shapes. Each test case passes a specific shape string to the method and expects the corresponding Unicode symbol to be printed on the console.

In the event of an unrecognized shape (e.g., "hexagon"), the expectation is that the method will print "I don't know this shape!!!" on the console.

This unit test scenario aims to validate the behavior of `printShapeInfo` independently, ensuring it correctly prints the corresponding Unicode symbol for recognized shapes and handles unknown shapes appropriately.

## **Integration Test Case for `identifyTetragon` and `printShapeInfo` Methods:**

```

# Test id - IT\_01

# Test purpose: Integration testing for 'identifyTetragon' and 'printShapeInfo' methods.

# Inputs: Four input values (a, b, c, d) representing one side of a proper tetragon for identifyTetragon.

# Testing procedure:

1. Call identifyTetragon method with input values (a, b, c, d) to identify the type of tetragon.

2. Pass the recognized shape (if successful) to the printShapeInfo method.

3. Capture the output printed on the console.

# Evaluation: Verifying if the correct symbol of the identified tetragon is printed.

# Expected behaviours and results:

- For valid inputs that represent a square: Expect to see '□' printed on the console.

- For valid inputs that represent a rectangle: Expect to see '△' printed on the console.

- For valid inputs that represent a trapezoid: Expect to see '♢' printed on the console.

- For invalid inputs: Expect to see 'I don’t know this shape!!!' printed on the console.

# Actual behaviours and results:

- Successful identification and printing of symbols for recognized tetragons.

- Proper handling of invalid inputs and printing appropriate messages for unknown shapes.

```

#Explanation:

This integration test case simulates the combined behavior of `identifyTetragon` and `printShapeInfo` methods in an integrated fashion. It performs a sequence where the identification of a tetragon shape by `identifyTetragon` leads to the printing of the corresponding symbol by `printShapeInfo`.

The test evaluates the correctness of symbol printing based on identified tetragons and checks the handling of invalid inputs or unrecognized shapes, aiming to ensure that both methods work together correctly and provide the expected behavior.