Magic Square

ASSIGNMENT 1

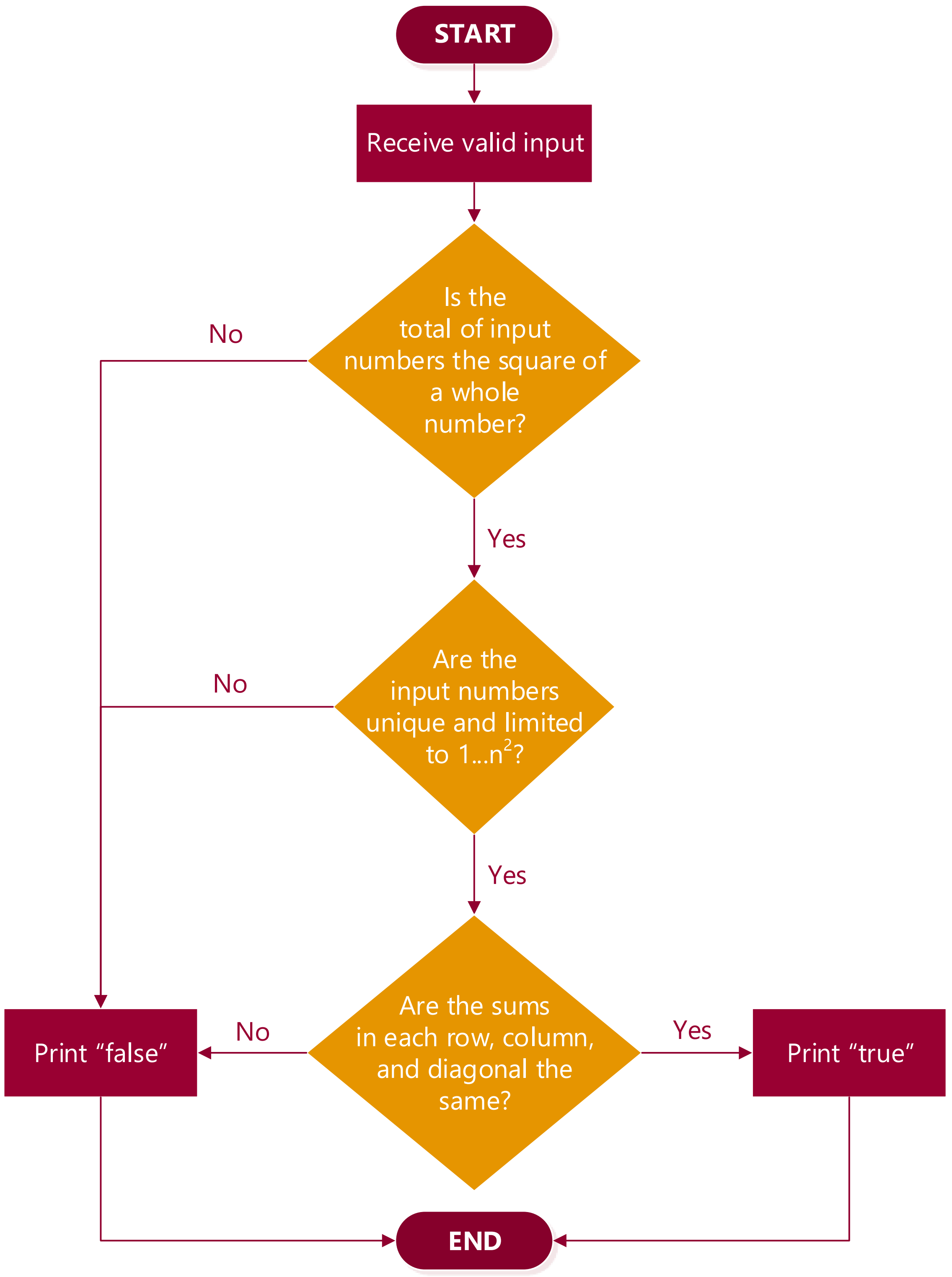
Marcinina Alvaran

SER 215 (ONLINE) | 2016 SPRING B

Table of Contents

1 DESIGN

1.1 Flowchart


START
Receive valid input
Is the total of input numbers the square of a whole number?
No
Yes
Are the input numbers unique and limited to 1...n²?
No
Yes
Are the sums in each row, column, and diagonal the same?
No
Yes
Print “false”
Print “true”
END

1.2 Method Descriptions

1.2.1 User instruction and information (introduction)

- Program header/title
- Describe input format and provide example

1.2.2 Input prompt and conversion (promptInput)

- **String** variable for user input (**input**)
- Ask for input using a brief message on a single line
- Convert input to **int**
- Store converted numbers in class **int[]** variable (**numberArray**)

1.2.3 Check if input is a Magic Square (isMagicSquare)

- Validate total number of inputs
- Validate numbers
- Validate sums
- Print **String** "true" if all magic square conditions are satisfied.
  Otherwise print "false".

1.2.4 Validation of input values total (validateTotal)

- **int** variable for total of number input values (**totalValues**)
- Compare totalValues to the square root integer of totalValues
  - If equal, return true.
    Otherwise, return false.

1.2.5 Validation of numbers (validateNumbers)

- Create an **int HashSet** with numbers 1…totalNumbers (**validNumbers**)
- Remove input matches from set
  - If no match is found, the number is either a repeat or out of bounds. Return false without checking remaining numbers.
  - Otherwise, return true.

1.2.6 Validation of sums (validateSums)

- **int** variable for target sum (**targetSum**)
- Store numbers into square format in **int[][]** variable (**squareArray**)
- Store sum of first row in targetSum
- Iteratively sum each row/column/diagonal while comparing to targetSum
  - If unequal, return false without calculating remaining sums.
    Otherwise, return true.

2 DESIGN REVIEW

i. Traceability

- Requirements addressed
  - ✓ User-friendly interface is addressed in **1.2.1**, **1.2.2**, and **1.2.3**.
  - ✓ Verification point 1 in specifications is addressed in **1.2.4**.
  - ✓ Verification point 2 in specifications is addressed in **1.2.5**.
  - ✓ Verification point 3 in specifications is addressed in **1.2.6**.
  - ✓ Output as described in specifications is addressed in **1.2.3**.
- Naming scheme
  - ✓ Validity-check support functions begin with "validate…" (ex. validateSums).
  - ✓ Validation return variables are named "validity".
- ✓ All functions and data are uniquely identified.

ii. Consistency

- ✓ The class is consistent with the information domain.
- ✓ The class is designed to check if an input represents a magic square.
- ✓ A standard design representation is used.
- ✓ A standard data usage representation is used.

iii. Completeness

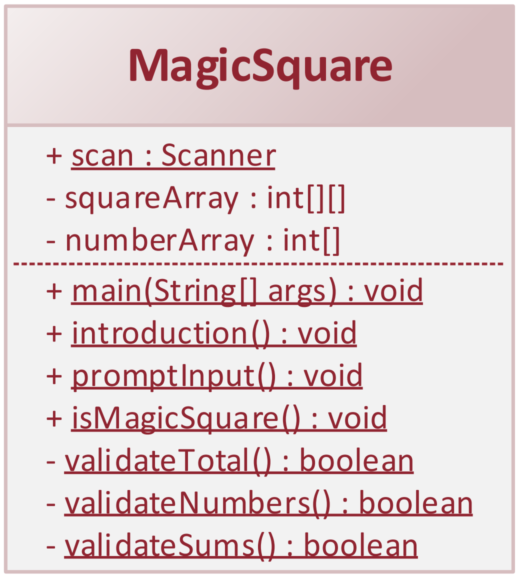
- ✓ Requirements are reflected in the software architecture.
- ✓ No referenced data were used.
- ✓ All defined data were used.
- ✓ No referenced modules were used.
- ✓ All defined modules were used.
- ✓ No interface was necessary.
- ✓ Maintainability was considered.
  - ✓ Scanner variable is a static class variable in case of changes involving further inputs.
  - ✓ All return statements are located at the bottom of the method to easily be located.

iv. Efficiency

- ✓ Data is grouped for efficient processing.
- ✓ The use of a two-dimensional array to store intermediate values was mentioned.
- ✓ Nearly all functions are independent.

3 SOFTWARE SIZE ESTIMATION

3.1 UML Diagram



3.2 Size Estimation Table

Module Description	Estimated Size (Lines of Code)
User instruction and information	8
Input prompt and conversion	10
Check if input is a Magic Square	6
Validate number input total	10
Validate numbers	11
Validate sums	14

TOTAL ESTIMATED SIZE: 59
(without Javadocs)

4 TESTING

Test Case 1: Invalid input – Incorrect number of input values
Input: 1 2 3 4 5
Output: false

Test Case 2: Invalid input – Duplicate numbers in input
Input: 1 2 2 4
Output: false

Test Case 3: Invalid input – Unequal sums
Input: 1 2 3 4
Output: false

Test Case 4: Valid input – All magic square conditions satisfied
Input: 8 1 6 3 5 7 4 9 2
Output: true

5 ACTUAL SOFTWARE SIZE

Module Description	Actual Size (Lines of Code)
User instruction and information	15
Input prompt and conversion	14
Check if input is a Magic Square	23
Validate number input total	15
Validate numbers	27
Validate sums	70

ACTUAL SOFTWARE SIZE: 164
(without Javadocs)

ESTIMATED SOFTWARE SIZE: 59
(without Javadocs)