**Design by Xiaqin Qiu:**

Classes needed to create: 3 classes altogether

**class Dice, class Rule, class Elimination.**

**In class Dice: (for Dice running to generate 2 random numbers)**

import java.util.Random for random numbers;

default constructor is necessary; use seta() and setb() methods to assgin the numbers

construct a random constructor;

**Attributes are:(2)**

int a; for number assignment 1

int b; for number assignment 2

**Methods are:(4)**

void seta(); for randoming a number from 1 to 6 and assign it to a, mutator

void setb(); for randoming a number from 1 to 6 and assign it to b, mutator

int geta(); to return a, accessor

int getb(); to return b, accessor

String toString(); to format the output

**In class Rule: (for all the running rules to the game)**

non-default constructor is necessary; 4 inputs as int a, int b, int s, String input / and use initiate(), seta(a), setb(b), sets(s), setInput(input) method to assign the values

**Attributes are:**

int a; for dice 1's number

int b; for dice 2's number

int s; for dices' sum

int score; for score calculated

String[] sign; string array for sign of tiles

String input; for inputting a character

boolean status; (initial=true) status for the outermost loop to decide whether to quit

boolean inputStatus; (initial=false) for innermost loop's input character's validation to decide whether to keep on the loop

**Methods are:**

void letsPlay(); check whether the numbers are qualified as whether they are equal or occuipied and decides whether let the User to input another character, and assigns the sign for the valid numbers and if the charater is not qualified or the number is occupied, tells tryAgain() and give hints. If a==b, the User can only sum it or tryAgain(). And use the numbers to ask whether the Value is valid or both dices and assign them if they are both qualified and ask whether the Sum is valid and assign it if it’s qualified

void initiate(); for initiating the sign[i]

void seta()/setb()/sets()/setInput()/setInputStatus()/setStatus(); to set the value, mutators

boolean getInputStatus(); to return the inputStatus value, accessor

boolean getStatus(); to return the status value, accessor

void quit(); use the Accessors to close the loops and calculate the score, and show() the last Play result with the score

void tryAgain(); for setting the inputStatus() by setInputStatus() to keep on the inner loop and give the "Try again." words

void show(); for show the tables of 1~12 and sign[i] and some align marks

boolean mark(int m); for judging whether a, b, s are qualified and give the true or false result

String toString(); format the object’s printing and for printing out the score (if someone wants to print the class in addition, show the score only; As all the things have been already printed out)

**In class Elimination: (the main method to achieve the running/playing)**

import java.util.Scanner for input;

Main method;

construct an object of Rule(a,b,s,input) named play;

construct a scanner named scan;

**Attributes are:**

String input=””; for input string

int a=0; for storing Dice 1’s number

int b=0; for storing Dice 2’s number

int s=0; for storing Dices’ sum number

**No methods;**

set up a loop to run the game and let the User to input the value again and again until stop();

in the loop, run dice each time and give the values to the “play” every time, and check the statuses to decide whether to end the loop which is letting the methods in “play” to decide.

**Collaboration:**

class Rule and class Dice;

**Composite:**

Elimination has a relationship of Rule;

Elimination has a relationship of Dice;

**Test Cases:**

|  |  |  |
| --- | --- | --- |
| TEST CASE | OUTPUT | TESTED? |
| Whether the Dice gives the right numbers | 1~6 each number | Yes |
| The input characters validation | both lower case & upper case of v/s/q are valid, and other input are invalid | Yes |
| Whether the align is right | right | Yes |
| Whether 2 same numbers for Value can be accepted | no | Yes |
| After input v/s, whether the valid number cross off the dots | cross off the right ones | Yes |
| If the number already occupied, new dice number tells "Try again." | Try again | Yes |
| Whether the input character does the correct things | v for value, s for sum, q for quit | Yes |
| Whether the last show is correct | 12 numbers, correct crosses, score | Yes |
| Whether the score is right | sum for all the dots' numbers | Yes |
|  |  |  |