Assignment 1

CS 1027
Computer Science Fundamentals II

Learning Outcomes

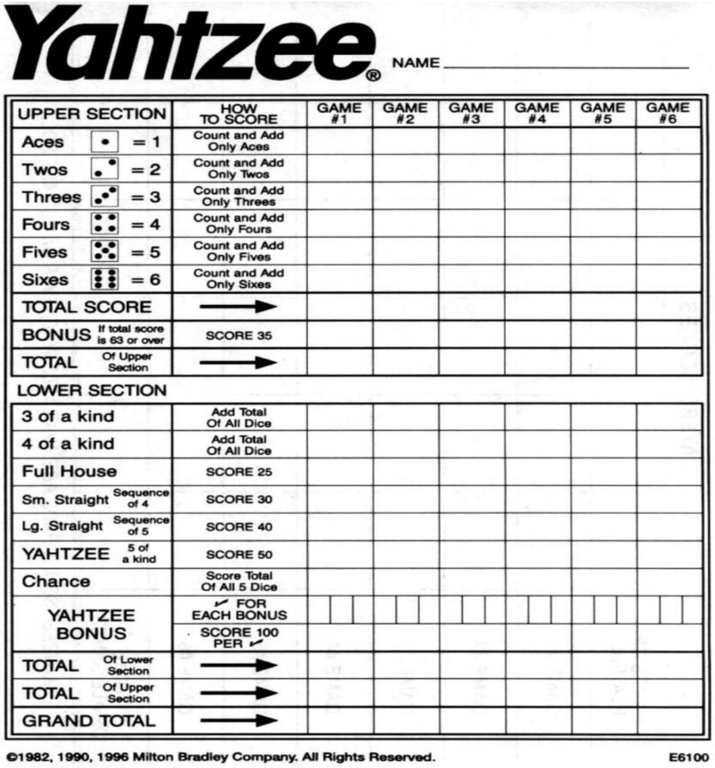
In this assignment, you will get practice with:

- Creating classes and objects of those classes
- Overloading constructors
- Implementing equals(), toString(), getters, and other methods
- Working with arrays
- Using loops and conditionals

Introduction

Yahtzee is a popular family game involving just 5 dice and a scorepad. On a player's turn, they roll the 5 dice together. Then they may choose to re-roll any number of those dice, and then again they can re-roll any number of the dice. After 3 rolls (or less if they choose), they cannot roll the dice anymore. They look at the values shown on the top of each of the 5 dice to score. This assignment focuses on how to compute a score from the values on the top of the 5 dice.

There are several ways to score by using the values on the 5 dice and typically looking for groups of the same number or a run of four or five consecutive values. Some of these are scored as the sum of all dice, some are the sum of a subset of the dice, and others are scored with a fixed amount of points. A picture of an actual Yahtzee scoresheet is pasted below.



Yahtzee® NAME ______________________

UPPER SECTION	HOW TO SCORE	GAME #1	GAME #2	GAME #3	GAME #4	GAME #5	GAME #6
Aces ⚀ = 1	Count and Add Only Aces						
Twos ⚁ = 2	Count and Add Only Twos						
Threes ⚂ = 3	Count and Add Only Threes						
Fours ⚃ = 4	Count and Add Only Fours						
Fives ⚄ = 5	Count and Add Only Fives						
Sixes ⚅ = 6	Count and Add Only Sixes						
TOTAL SCORE	→						
BONUS If total score is 63 or over	SCORE 35						
TOTAL Of Upper Section	→						
LOWER SECTION							
3 of a kind	Add Total Of All Dice						
4 of a kind	Add Total Of All Dice						
Full House	SCORE 25						
Sm. Straight Sequence of 4	SCORE 30						
Lg. Straight Sequence of 5	SCORE 40						
YAHTZEE 5 of a kind	SCORE 50						
Chance	Score Total Of All 5 Dice						
YAHTZEE BONUS	✓ FOR EACH BONUS						
	SCORE 100 PER ✓						
TOTAL Of Lower Section	→						
TOTAL Of Upper Section	→						
GRAND TOTAL	→						

©1982, 1990, 1996 Milton Bradley Company. All Rights Reserved. E6100

Notice the 6 scoring options in the "UPPER SECTION" in which the scores are calculated as the sum of the dice showing a specified value. Notice also the 7 scoring options in the "LOWER SECTION" which include: 3 of a kind (3 dice showing the same value), 4 of a kind (4 dice showing the same value), Full House (3 of a kind and a pair), Small Straight (sequence of 4 dice with consecutive values), Large Straight (sequence of 5 dice with consecutive values), YAHTZEE (5 dice showing the same value), and Chance (nothing special in the dice). Ignore the Bonus and Total rows in the scoresheet.

In this assignment, given the values on the top of 5 dice you have to determine the possible scores of all 13 options listed here and store them in an array. For some of these scoring options, a certain condition has to be met (i.e. for 3 of a kind, you have to determine if there are 3 dice showing the same value) in order to include the corresponding score in the array. For others, like the first 6 options, there aren't any conditions to be met; rather you just have to sum the specific subset of dice to calculate the score for that option. The order of the dice does not matter for scoring! The score options are summarized in the following table. The table also includes an example of each score option; the numbers shown in blue text represent the dice that are being used to compute the score for that row.

Option	Description	Scoring	Example
Aces	Dice showing 1	Sum of the dice showing 1	{1, 1, 1, 4, 5} = 3
Twos	Dice showing 2	Sum of the dice showing 2	{1, 2, 2, 5, 6} = 4
Threes	Dice showing 3	Sum of the dice showing 3	{1, 3, 3, 4, 5} = 6
Fours	Dice showing 4	Sum of the dice showing 4	{2, 3, 4, 6, 6} = 4
Fives	Dice showing 5	Sum of the dice showing 5	{1, 2, 5, 5, 5} = 15
Sixes	Dice showing 6	Sum of the dice showing 6	{2, 4, 4, 6, 6} = 12
3 of a kind	3 dice showing the same value	Sum of all 5 dice	{2, 4, 4, 4, 5} = 19
4 of a kind	4 dice showing the same value	Sum of all 5 dice	{3, 3, 3, 3, 6} = 18
Full House	3 dice of a value and 2 dice of another value	Fixed score of 25	{1, 1, 1, 5, 5} = 25
Small Straight	4 dice with consecutive values	Fixed score of 30	{2, 3, 3, 4, 5} = 30
Large Straight	5 dice with consecutive values	Fixed score of 40	{1, 2, 3, 4, 5} = 40
Yahtzee	5 dice showing the same value	Fixed score of 50	{2, 2, 2, 2, 2} = 50
Chance	(No pattern)	Sum of all 5 dice	{1, 3, 3, 5, 6} = 18

For example,

- suppose we have the given dice: {5, 3, 5, 4, 2}.
- the scoring options would be: [0,2,3,4,10,0,0,0,0,30,0,0,19].

The reasoning for these scores: 0 ones, 1 two (2 pts), 1 three (3 pts), 1 four (4 pts), 2 fives (10 pts), 0 sixes, no 3 of a kind, no 4 of a kind, no full house, small straight (30 pts), no large straight, no Yahtzee, and the Chance (19 pts).

Provided files

The following is a list of files provided to you for this assignment. **Do not alter these files**.

- RandomNumber.java
- TestGame.java

Classes to Implement

For this assignment, you must implement two Java classes: Dice and Yahtzee. Follow the guidelines for each one below.

In all these classes, you can implement more private (helper) methods, if you want to, but you may not implement more public methods. You may not add instance variables other than the ones specified below nor change the variable types or accessibility (i.e. making a variable public when it should be private). Penalties will be applied if you implement additional instance variables or change the variable types or modifiers from what is described here.

Dice.java

This class represents a single six-sided die (dice) that will be used in the game.

The class must have the following private variables:

- value (int)

The class must have the following public methods:

- public Dice() [constructor]
    - Initialize value to -1
- public Dice(int) [constructor]
    - Initialize value to the given argument
- public void roll()
    - Use the provided class RandomNumber to generate a number between 1 and 6 (inclusive) and set the value to that number
- public int getValue()
    - Returns the dice value

Yahtzee.java

This class represents the Yahtzee game in which there are 5 rolled dice and the scoring is performed. This class is much longer and more complex than the Dice class.

The class must have the following private variables:

- dice (Dice[ ])

The class must have the following public methods:

- public Yahtzee() [constructor]
  - Initialize the dice array and roll them for random values
- public Yahtzee(Dice[ ]) [constructor]
  - Initialize the dice array with the given argument
- public int[ ] getValueCount()
  - Count how many dice show each of the possible values from 1-6 and record all of them in an int array. Return this array with the 6 counters.
  - Entry 0 in this array is for the number of ones, entry 1 in the array is for the number of twos, and so on.
  - i.e. if the 5 dice are: {1, 4, 2, 1, 5}, this method should return the array: [2, 1, 0, 1, 1, 0] because there are 2 ones, 1 two, 0 threes, 1 four, 1 five, and 0 sixes.
- public int[ ] getScoreOptions()
  - Create an int array with 13 elements, to record all the possible scores for the dice in the instance variable. These 13 scores must be ordered exactly as described in the introduction (beginning with the "Sum of 1s" in index 0 and ending with the "Yahtzee" (5 of a Kind) in index 12).
  - Hint: you may want to create one or more private helper methods that can be called from this method. You also may want to use the getValueCount() method.
- public int[ ] score()
  - Call getScoreOptions() and then determine the maximum value from the array of possible scores, and the index at which the maximum is found (if the maximum value appears more than once, find the smallest index where the maximum value appears)
  - Return an int array containing 2 values: the maximum value and then the corresponding index of that value
- public boolean equals(Yahtzee)
  - Compare the given Yahtzee object from the argument with the "this" object to see if they are equal. Consider equality to be the same 5 dice but in any order.
  - i.e. The dice: {2, 6, 5, 1, 2} is considered equal to the dice: {5, 1, 2, 2, 6}.
- public String toString()
  - Return a string of the dice values formatted this way: "Dice: {3, 5, 1, 1, 2}"

Marking Notes

Functional Specifications

- Does the program behave according to specifications?
- Does it produce the correct output and pass all tests?
- Are the classes implemented properly?

- Does the code run properly on Gradescope (even if it runs on Eclipse, it is up to you to ensure it works on Gradescope to get the test marks)
- Does the program produces compilation or run-time errors on Gradescope?
- Does the program fail to follow the instructions (i.e. changing variable types, etc.)

Non-Functional Specifications

- Are there comments throughout the code (Javadocs or other comments)?
- Are the variables and methods given appropriate, meaningful names?
- Is the code clean and readable with proper indenting and white-space?
- Is the code consistent regarding formatting and naming conventions?
- Submission errors (i.e. missing files, too many files, etc.) will receive a penalty of 5%
- Including a "package" line at the top of a file will receive a penalty of 5%

Remember you must do all the work on your own. Do not copy or even look at the work of another student. All submitted code will be run through similarity-detection software.

Submission (due Monday, February 7, 2022 at 11:55pm ET)

Assignments must be submitted to Gradescope, not on OWL. If you are new to this platform, see these instructions on submitting on Gradescope.

Rules

- Please only submit the files specified below.
- Do not attach other files even if they were part of the assignment.
- Do not upload the .class files! Penalties will be applied for this.
- Submit the assignment on time. Late submissions will receive a penalty of 10% per day.
- Forgetting to submit is not a valid excuse for submitting late.
- Submissions must be done through Gradescope. **If your code runs on Eclipse but not on Gradescope, you will NOT get the marks! Make sure it works on Gradescope to get these marks.**
- You are expected to perform additional testing (create your own test harness class to do this) to ensure that your code works for other dice combinations as well. We are providing you with some tests but we may use additional tests that you haven't seen before for marking.
- Assignment files are NOT to be emailed to the instructor(s) or TA(s). They will not be marked if sent by email.
- You may re-submit code if your previous submission was not complete or correct, however, re-submissions after the regular assignment deadline will receive a penalty.

Assignment 1

CS 1027
Computer Science Fundamentals II

Files to submit

- Dice.java
- Yahtzee.java

Grading Criteria

Total Marks: [20]

Functional Specifications:

[1] Dice.java

[3] Yahtzee.java

[13] Passing Tests

Non-Functional Specifications:

[1] Meaningful variable names, private instance variables

[1] Code readability and indentation

[1] Code comments