Jared Weyer

CPSC 224, Spring 2020

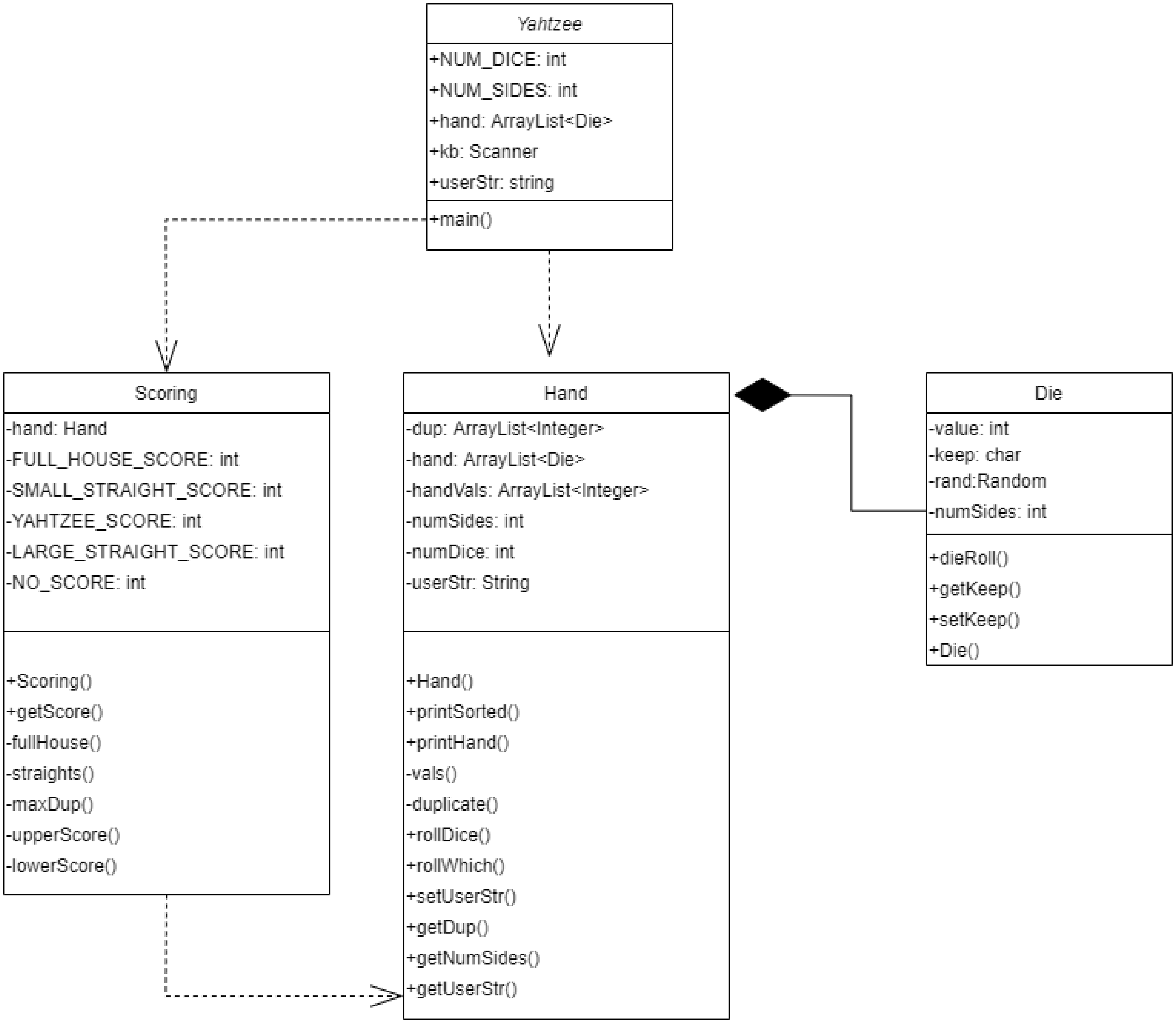
Programming assignment #2

February 13, 2020

NOTE

This file has not been modified for the second homework as there has not been any major changes to the program and thus no changes to the UML diagram

The goal of the Yahtzee program is to create a program in which a user can play a simplified virtual version of the game Yahtzee. This simplified version should allow the user to roll a hand of die; the user should be able to choose which die in the hand that he wants to roll. After the user uses all his rolls/turns the program should output the scores in which the hand would receive on the Yahtzee scorecard. The user should also be able to play as many games as he wants. In the design of the Yahtzee program I implemented four classes: the main Yahtzee Class, the Hand Class, the Die Class, and the Scoring Class (See the UML diagram below for a description of these classes). The program design was implemented so that a user will have one hand object which contains a certain number of die objects (for Yahtzee there will be 5 die objects). The main class will allow the user to interface with the Hand object. These interactions will be limited to the user being able to control which of the die in his hand that he/she wants to roll. After the users turns are exhausted the Scoring class will be used to determine the different scores in which the final hand has earned. These scores will be determined by giving the Scoring class access to the hand object and the Hand class; further, the scores will be based off the rules listed in the Yahtzee board game. The main challenge I faced when creating the Yahtzee program was figuring out the right number of classes to implement. In my first iteration of the program design I incorporated too many classes which were all interdependent on one another. This dependency created an overcomplicated code in which I decided to completely delete and start from scratch. In the end I found that using four classes was the right balance between overcomplication and classes that had too much functionality (god classes). In retrospect doing more design work on the program before I started to code would have saved me a lot of time. However, I leaned quite a bit about OOP programming by just jumping into the coding and learning from my mistakes. Below is the UML diagram associated with the Yahtzee program:


Yahtzee
+NUM_DICE: int
+NUM_SIDES: int
+hand: ArrayList<Die>
+kb: Scanner
+userStr: string
+main()
Scoring
-hand: Hand
-FULL_HOUSE_SCORE: int
-SMALL_STRAIGHT_SCORE: int
-YAHTZEE_SCORE: int
-LARGE_STRAIGHT_SCORE: int
-NO_SCORE: int
+Scoring()
+getScore()
-fullHouse()
-straights()
-maxDup()
-upperScore()
-lowerScore()
Hand
-dup: ArrayList<Integer>
-hand: ArrayList<Die>
-handVals: ArrayList<Integer>
-numSides: int
-numDice: int
-userStr: String
+Hand()
+printSorted()
+printHand()
-vals()
-duplicate()
+rollDice()
+rollWhich()
+setUserStr()
+getDup()
+getNumSides()
+getUserStr()
Die
-value: int
-keep: char
-rand:Random
-numSides: int
+dieRoll()
+getKeep()
+setKeep()
+Die()