Josh Staples
December 14, 2021
Computer Programming I
Final Project Design Documentation

Craps Game

Algorithm:

Separate method to roll two dice and return the sum of the results

- Important to "roll" two separate 6-sided dice as opposed to a random number between 1-12 because the odds of getting a 7 in each scenario is different (1/12 on 1 1-12 roll vs. 1/6 for 2 1-6 rolls)

Roll the dice for the user, and if the user doesn't immediately get craps (2, 3, 12) or a natural (7, 11), "roll" the dice for 2 or 3 other "computer players" and check the same for them.

All players standing get assigned their "point" as the value they initially rolled, each player continues to roll until they get their "point" or craps

Create array as a sort of "leaderboard" where the player can see who lost first to last

Display leaderboard to user after there is only one person still rolling

---

Method Signatures:

CrapsGame Class:

main(String[] args) – main method
startGame(Player[] crapsPlayers) – runs the entire craps game, calls other methods
putFirstAvaiable(Player crapsPlayer) – puts the player into the first available leaderboard spot
putLastAvailable(Player crapsPlayer) – puts the player into the last available leaderboard spot
printLeaderboard() – prints the entire leaderboard of players
arrayContains(Player[] playerList, Player crapsPlayer) – checks if a player is in the array
getYesOrNo() – gets a simple yes or no response from the user
getSumOfDice() – rolls two independent dice and returns the sum of their rolls
checkRoll(int roll) – returns 1 if roll is natural, 0 if its nothing, -1 if its craps
getUserName() – asks the user for their name and returns the result

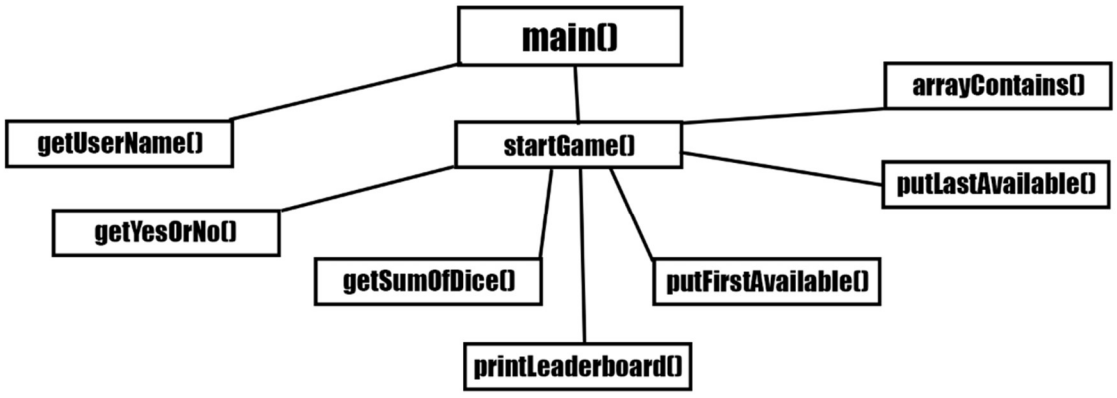
---

Josh Staples
December 14, 2021
Computer Programming I
Final Project Design Documentation

Player Class:

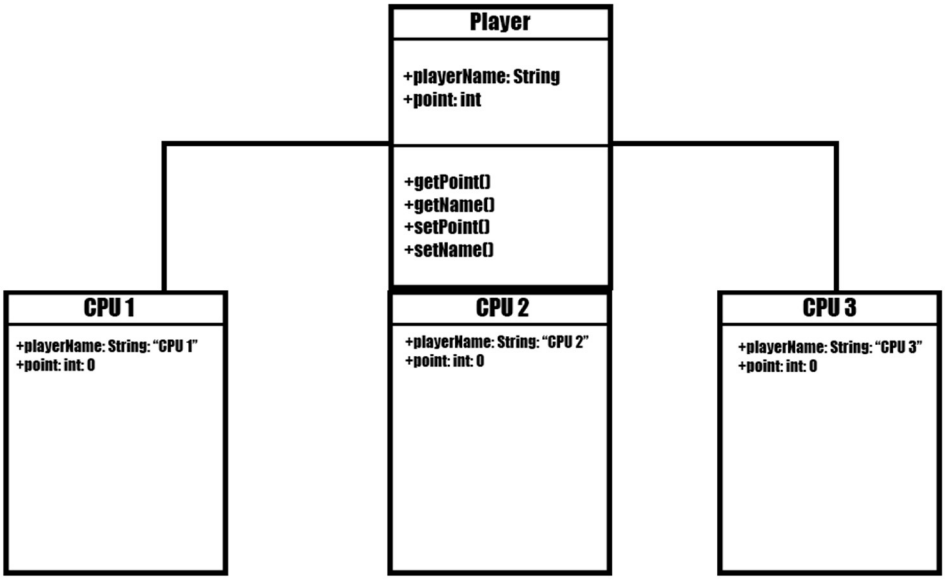
Player(String playerName) – constructor for the player
getPoint() – returns the user's point
getName() – returns the user's name
setpoint() – sets the user's point
setName() – sets the user's name

Module Structure Chart:



Josh Staples
December 14, 2021
Computer Programming I
Final Project Design Documentation

UML Diagram:



Test Set/Plan:

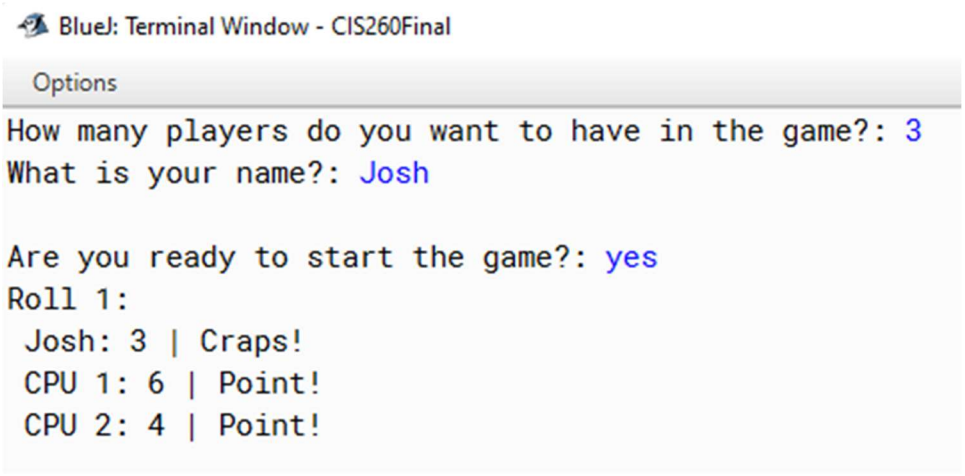
To help debug, the program will have println statements that print the initial rolls on the dice and check that everything adds up properly

Roll 1	Roll 2	Point	"turn"	Result
Adds to 2, 3, or 12		Not assigned	any	Craps, user loses
Adds to 7 or 11		Not assigned	1	Natural, user wins
Adds to 4, 5, etc. (not craps or natural)		Assigned to sum	1	Point set to sum, turn ends
Adds to 4, 5, etc. (not craps)		Same as roll	2+	Same as point, user wins

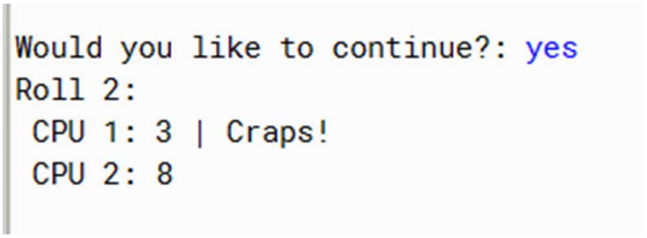
Each situation applies to the "computer players" as well, although user used as an example for test – very hard to define too many circumstances as the game seems relatively easy and basic (at least the simplified version)

Josh Staples
December 14, 2021
Computer Programming I
Final Project Design Documentation

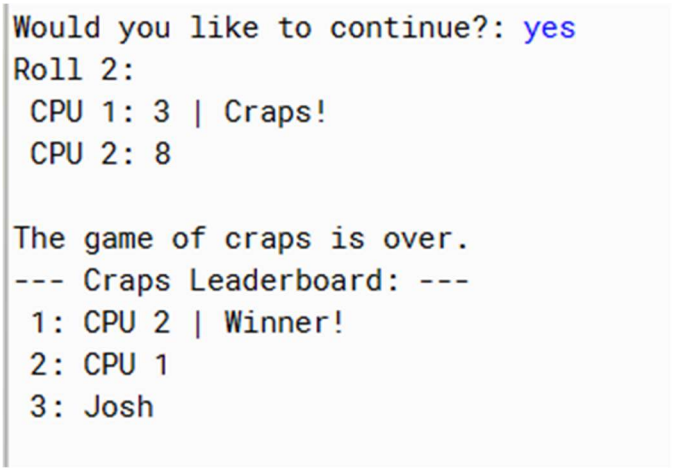
Sample Outputs:



*user is removed from the game after rolling craps/natural



*if a player rolls craps on any turn they're removed



Josh Staples
December 14, 2021
Computer Programming I
Final Project Design Documentation

*naturals remove players, craps and natural can happen on the same roll

```
Are you ready to start the game?: yes
Roll 1:
 Josh: 11 | Natural!
 CPU 1: 8 | Point!
 CPU 2: 9 | Point!
 CPU 3: 12 | Craps!
```

*rolling the same number that is your point gives you the win

```
Are you ready to start the game?: yes
Roll 1:
 Josh: 10 | Point!
 CPU 1: 6 | Point!
 CPU 2: 9 | Point!
 CPU 3: 6 | Point!

Would you like to continue?: yes
Roll 2:
 Josh: 10 | Rolled Point!
 CPU 1: 12 | Craps!
 CPU 2: 9 | Rolled Point!
 CPU 3: 7
```

* first player to roll natural/their point is first place, followed by the proper order

```
Would you like to continue?: yes
Roll 2:
 Josh: 10 | Rolled Point!
 CPU 1: 12 | Craps!
 CPU 2: 9 | Rolled Point!
 CPU 3: 7

The game of craps is over.
--- Craps Leaderboard: ---
 1: Josh | Winner!
 2: CPU 2
 3: CPU 3
 4: CPU 1
```