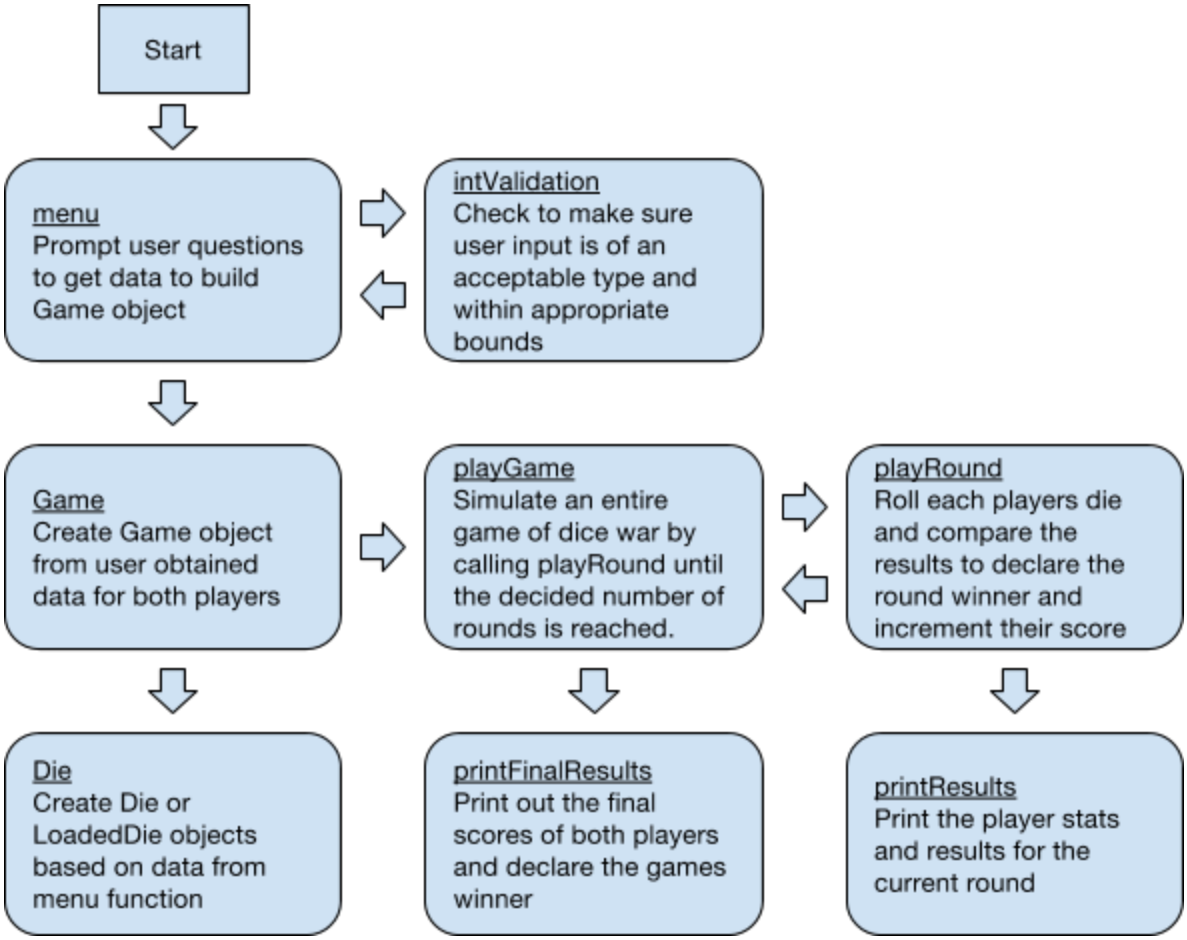
Joel Huffman
1/26/2018
Lab 3 Reflection

Design:
Below is the flowchart I used for the general outline design for my program.



Changes made during implementation:
I originally was going to have the Die and LoadedDie objects created before the Game object was created and have the Game object receive pointers to those Die objects, but after some reworking I opted for creating the Die objects from the Game class to simplify the overall design and maintain logical flow.
I also ended up implementing a while loop in the menu function to allow the user to keep playing dice game until they were ready to quit. This allowed me to run my test cases faster for the debugging process and I'm sure will help expedite the grading of the assignment to make things a little easier for the TAs.

Problems encountered during implementation and how they were solved:
I had an issue in which I was trying to call LoadedDie's roll() function, but Die's roll() function was being called instead. This was bad because it rendered my Die and LoadedDie objects functionally equivalent. I found the solution by implementing the virtual keyword into the

parent/base class' roll() function. This allowed both roll() functions to be called appropriately from their corresponding class objects.
I also ran into a problem with scope and the Die objects. They were being created within the Game object, but their scope was being limited to the function they were created in. I solved this issue by using the using the new keyword when defining them so they could be referenced throughout my Game object. Dynamically allocating memory for these also required me to make sure to deallocate and delete the pointers once the game concluded.

Test Plan:

Test Case	Input	Expected Outcome	Observed Outcome
Loaded die vs regular die, Same # of sides.	P1 dieType: Loaded P2 dieType: Regular P1 dieSides: 6 P2 dieSides: 6 Rounds: 100	P1 score: ~60 P2 Score: ~40 P1 wins! P1 should win by a large margin	P1 score: 65 P2 score: 21 P1 wins!
Regular die vs regular die, Same # of sides.	P1 dieType: Regular P2 dieType: Regular P1 dieSides: 10 P2 dieSides: 10 Rounds: 100	P1 score: ~50 P2 Score: ~50 It's a draw Scores should be close	P1 score: 47 P2 score: 41 P1 wins!
Loaded die vs loaded die, Same # of sides.	P1 dieType: Loaded P2 dieType: Loaded P1 dieSides: 99 P2 dieSides: 99 Rounds: 100	P1 score: ~50 P2 Score: ~50 It's a draw! Scores should be close	P1 score: 41 P2 score: 34 P1 wins!
Loaded die vs regular die, loaded die has double the sides as regular.	P1 dieType: Loaded P2 dieType: Regular P1 dieSides: 20 P2 dieSides: 10 Rounds: 300	P1 score: ~250 P2 Score: ~50 P1 wins! P1 should win by a landslide	P1 score: 266 P2 score: 32 P1 wins!
Loaded die vs regular die, loaded die has halve the sides as regular.	P1 dieType: Loaded P2 dieType: Regular P1 dieSides: 10 P2 dieSides: 20 Rounds: 100	P1 score: ~40 P2 Score: ~60 P2 wins! P2 shouldn't win by a large margin	P1 score: 34 P2 score: 59 P1 wins!