Program name: Dice ‘War’ Game
Program purpose: Two players choose between a regular and loaded die. Each round, both players roll and the one with the highest number wins. If both numbers are the same, it is a draw.

Reflection:

My original design detailed the roll functions for each die and the intended data members which were the sides and current roll value. I detailed a separate input validation function, because I knew having something reusable would be great for future projects. I put checking for floats and integers as a main part of the validation function. I detailed the a menu and game play function for the game class. Mostly, I focused on the output of these sections.

I originally thought I would use an array to keep track of the die sides and values, but I realized this was much more complicated than it need to be. I simplified to simply keeping track of how many sides for each die and focusing on the random roll. I mostly followed my design, but I ended up having more variables in the game class than I anticipated. I still don’t like this aspect of the design but it was the most straightforward way I could think of to do things.

My main problem during implementation was I forgot to make the loaded die class inherit from the die class. As a result, I ended up redesigning a lot of code halfway through. Additionally, I originally was keeping score in the die class, but I realized I expected only 1 player to be able to have a loaded die. After reading over the guidelines again, I realized it was possible for both players to pick a loaded die or a regular die. I again had to redesign a lot of code. Next time, I need to triple check the requirements before starting to code.

Test plan:

Test	Input	Test Target	Expected Output	Actual Output
Draw results in no points	Dice roll the same number	Game::play()	No points added for either player	No points added for either player
Player with highest win points	One player rolls higher than another	Game::play()	Player with higher roll gets 1 point at it	Player with higher roll gets 1 point at it
Prints out stats every round	6 rounds	Game::play()	Prints out 6 times. Each print out has number of sides for die, type of die for players, score of players and roll of players.	Prints out 6 times. Each print out has number of sides for die, type of die for players, score of players and roll of players.
Correct number of rounds	6 rounds	Game::setPlayersAndRolls()	6 print outs	6 print outs

Player gets correct die	Player 1 is loaded, player 2 is regular	Game::play()
		Game::menu()
			Player 1 prints out with loaded
			Player 2 prints out with regular
				Player 1 prints out with loaded
				Player 2 prints out with regular

INPUT VALIDATION				
Each prompt rejects char (6 prompts)	1. hgjhjf	Game::menu()	1. Error asking for integer	1. Error asking for integer
Each prompt rejects floats	1. 1.11	Game::menu()	1. Error asking for integer	1. Error asking for integer
Rounds prompt rejects 0 or below	1. -1 2. 0 3. 2	Game::menu()	1-2. Error asking for value in range 3. Regular game play	1-2. Error asking for value in range 3. Regular game play
Sides prompts rejects 0 or below	1. -1 2. 0 3. 2	Game::menu()	1-3. Error asking for value in range	1-3. Error asking for value in range
Prompt asking to play accepts between 0 and 1	1. 0 2. 2 3. -1	Game::menu()	1. Regular game play 2-3. Error asking for value in range	1. Regular game play 2-3. Error asking for value in range
Prompt asking to play again accepts between 0 and 1	1. 0 2. 2 3. -1	Game::menu()	1. Regular game play 2-3. Error asking for value in range	1. Regular game play 2-3. Error asking for value in range
Prompt asking to choose die accepts between 0 and 1	1. 0 2. 2 3. -1	Game::menu()	1. Regular game play 2-3. Error asking for value in range	1. Regular game play 2-3. Error asking for value in range
FLIP				
Valgrind: no segmentation faults	valgrind lab3	Whole program	1. No errors	ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
Valgrind: No memory leaks or	valgrind --tool=memcheck --leak-check=full --show-leak-kinds=all --track-origins=yes ./lab3	Whole program	1. No errors	ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
Complies on school server	1. g++ -std=c++0x main.cpp ant.cpp menu.cpp -o ant 2. ./lab3	Whole program	1. nothing 2. First prompt (# of rows in board)	1. nothing 2. First prompt (# of rows in board)