Submission Form

Please ensure you fully complete all FOUR questions on this form before submitting your assignment.

1. Please list the programming constructs you have used.

If statements

else if statements

else statements

While loops

2. Describe in simple language how your program works.

My program is intended to calculate the average number of rolls required to get 6 on a dice, it generates a random number between 1 and 6 to imitate the rolling of a single dice, printing the rolls as it goes and keeping track of the amount of rolls it took until it reaches a 6. it then prints the number of rolls it took in total. It then proceeds to calculate the average number of rolls by dividing the number of rolls it took by the amount of dice that were rolled, which is one in the first instance. The user can roll again using the same method by entering a positive number and the program will then roll the dice and recalculate the average using the new amount of rolls it took added to the last amount and with the number of dice increased by 1, the user can repeat this action until they decide to quit by entering 0. If they quit it will display the calculated average number of rolls to get a 6 and the number of 6's that is based upon (which is the same as the number of dice that were rolled). If the user gives an input that is not a number or not positive/zero and is therefore an invalid input, the program will ask the user for a valid input until one is recieved.

For a more comprehensive breakdown:

the program will generate a random number.

Check if number is not equal to 6, if it is not 6 then continue iterating the loop.

Send the generated number to display every time one is generated

Add one to dicerolls variable every time loop iterates.

Terminate loop once six is reached.

Add one to the dicenum variable

Send the dicerolls variable to display

Calculate the average number of dice rolls required to get a 6 using the dice rolls and dice number.

Ask the user if they wish to continue or terminate the program

if the user enters 0 then the program will terminate which will send the calculated average number of rolls to get a 6(average) and the number of sixes that is based upon (dicenum) to display.

If the user enters a number that is not an integer or is negative they will be asked again for the answer until a valid number is provided.

If the user enters a positive number the program will run again, retaining the data from the previous iteration.

3. Please describe below any additional features that you've included in your programme, if none what would you add if you had more time?

I have added some fairly significant additional features beyond the brief, largely for my own debugging purposes. I added a feature such that it prints the output of each roll, and then a statement at the end that tells you the total number of rolls it took to get the 6. This was added so I could check the maths of working out the average, with the printing of each roll added so I could ensure that the dicerolls variable was counting accurately, as well as the fact it's just interesting to see how lucky the roll has been.

I have added greater input validation to the program than was specified, I have ensured that the value entered cannot be less than the specified 0 that is used to exit the program and have verified the user input is not an invalid data type, each with a respective message. This is why an additional while loop was used to return to the point where an user input is entered.

If I had more time it does have a cosmetic issue I would like to fix - which is that if the user enters a number before entering an invalid data type, the program runs as it should as if a positive number was entered first, but it also displays the message as if cin were invalid, which is technically also true as an invalid data type was entered during the input. I would ideally like it to not print this message though this is a very specific break that requires the user to heavily misunderstand and ultimately does not functionally impact the program beyond printing an additional line, it would still be something I'd want to modify if I were to make the program completely foolproof.

I would perhaps add a new variable so that I could print the number of rolls it took each time the program was run next to the total amount of dice rolls, even though it's very easy to work out based on the last total number VS current total number or even by counting the amount of numbers displayed.

If I required to use this program often I would also perhaps add the ability to reset the current average/dicenum/dicerolls variables to 0, so that if you wanted to calculate multiple completely different averages you could do so without having to rerun the program completely.

4. Please paste your code into the box below, ensuring it has been formatted correctly

```
#include <iostream>;
#include <cstdlib>;
#include <ctime>;
using namespace std;
int main()
{
    srand(time(0));
    double answer = 0;
    double dicerolls = 0.0; // for accuracy of average use real numbers.
    double dicenum = 0.0;
    double average = 0.0;

    while (true) {
        int roll = 0;
        while (roll != 6) {
            roll = rand() % 6 + 1;
            cout << roll << endl;
            dicerolls++; // counting the number of iterations until a 6 is reached
        }
        dicenum++; //counting the total number of times the user has chosen to roll the dice

        cout << "Total number of rolls taken: " << dicerolls << endl;

        average = dicerolls / dicenum; //calculate average

        while (true) {
            cout << dicenum << " dice rolled, would you like to continue? Exit(0) Continue(PosNum)";
            cin >> answer;
            cout << endl;
            if (!cin) { //any non-double data type will set this off, which for our purposes is any
non-numeric value
                cout << "That is not a valid number, please enter only a number." << endl;
                cin.clear(); //reset cin so it doesn't read as incorrect forever.
                cin.ignore(numeric_limits<streamsize>::max(), '\n'); //clear entire input buffer
so a new input can be taken
                continue;
            }
            else if (answer == 0) {
                cout << "The average number of rolls to get a 6 was " << average << endl;
                cout << "Number of sixes that average is based upon is " << dicenum << endl;
                return 0;
            }
            else if (answer > 0) {
                break; // terminate current while and return to encompassing while loop.
            }
            else {
                cout << "That is not a valid number, please enter one." << endl;
                continue;
            }
        }
    }
}
```