LAB 4 REQUIREMENTS

Problem:

In this lab, you will be using Visual Studio to implement the classic game of Craps. Craps is a game of numbers and dice. The challenge is that when the user rolls the dice there are two sets of rules to follow depending on whether the roll is the users first roll to set the point/goal or it is one of potentially many rolls after the point/goal has been set. The rules are detailed below.

PART A

The rules for the Craps game are below.

The Rules

There are a few different variations of Craps. In this version, the player rolls two dice, and the spots on the dice are added together. The player wins on the first roll if the dice show a total of 7 or 11. The player loses if the total is 2, 3, or 12. If another number is thrown on the first roll - for example, a 10 - that number becomes the point, or the goal of subsequent tosses. The player continues to roll the dice until the point or a 7 appears. Throwing a total equal to the point means the player wins and throwing a 7 means the player loses. This is the version of Craps usually played in Casinos.

Some examples:

First Toss:	3	Immediate Loss

First Toss:	11	Immediate Win

First Toss:	10	<This is the point or goal>
Second Toss:	9	
Third Toss:	4	
Fourth Toss:	10	Win

First Toss:	10	<This is the point or goal>
Second Toss:	9	
Third Toss:	7	Loss

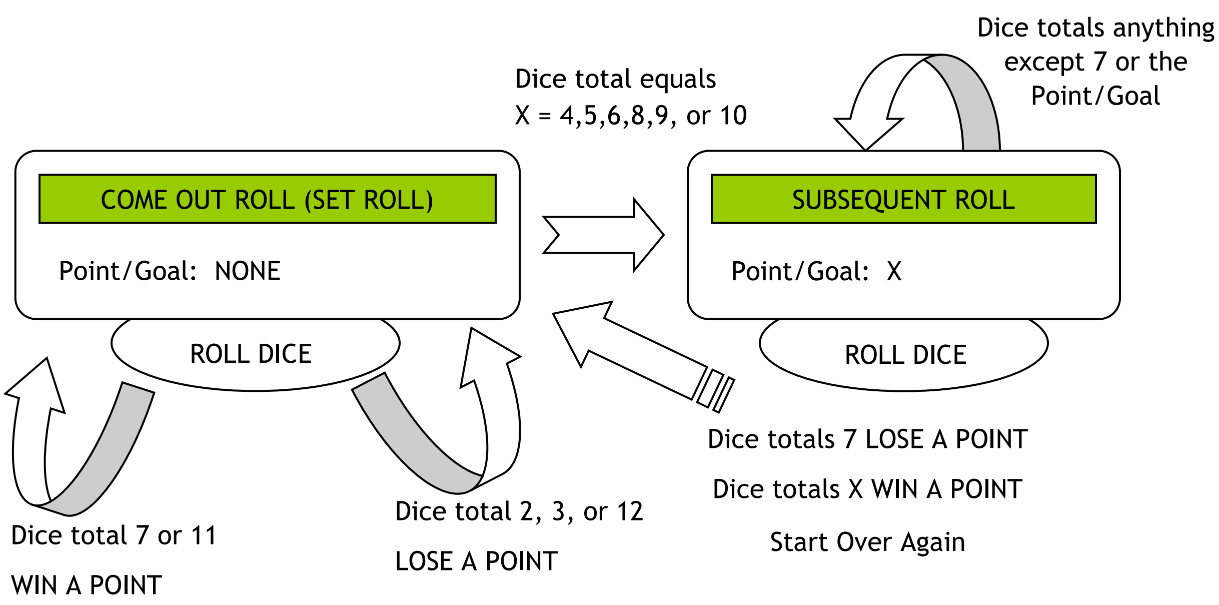
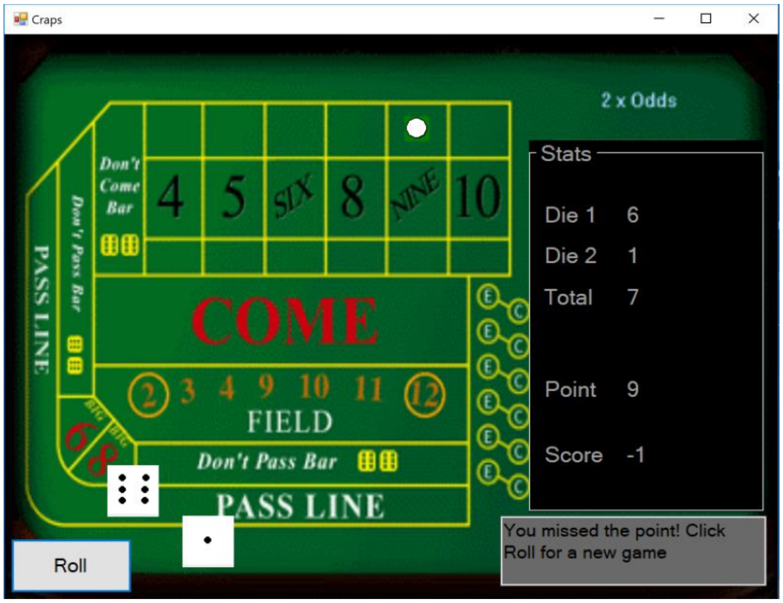


Figure 1: A Flowchart for the Game of Craps

In this program, you will need to use the Random class to randomly generate 2 numbers to determine what the user rolled on each die (2 dice total).

When the player clicks on the 'Roll the dice' button the computer will randomly select two numbers. Using these randomly select numbers, it will display two dice representing the correct numbers that were chosen. It will also display the numbers for each die and the total. After rolling the dice, the game status will be displayed by the placement of the puck. A black puck indicates the game is in the *come out roll (set roll)* state. A white puck indicates a *subsequent roll* and the point value (or goal). A message indicating a win, loss, or other relevant information should also be displayed.

This game should keep score, if a player wins, one should be added to the score and if the player loses, one should be subtracted from the score. The score can be negative. Below is a sample screen shot. You don't have to implement the Craps game exactly as shown as long as it has all of the components described in the lab.



Stats

Die 1 6

Die 2 1

Total 7

Point 9

Score -1

You missed the point! Click Roll for a new game

In the example shown above, on a subsequent roll, the user rolled a seven (one and six). This means that the Point value is not matched before a roll of seven and you lost. The program should be set to the first roll state for the next roll of the dice.

Setup the Form for your Craps game.

For the die images, copy the images from the web site for each possible die face (1, 2, 3, 4, 5, 6) and paste the image into a PictureBox control as demonstrated in class. For each die, draw six different PictureBox controls in C# .NET. Use these images for the die faces to appear in the PictureBox controls.

PART B

Finish creating your game by writing the code for the 'Roll the dice' button and the 'Exit' button. You will need to use a Form-level variable (field) to store the Point value (or goal). Remember, the Point value is not the same as the score, the terminology may seem confusing. If it is confusing, reread the gaming directions given above. You will also need a form-level variable (field) to keep track of the 'State' of the game. The game is either in *First Roll* or in *Roll Again*. In *First Roll*, the game will need to roll two dice and establish whether the user is an instant winner, an instant loser, or establish a Point (a number that is the goal to roll again before rolling a 7). In *Roll Again*, the game is rolling again – it already has established a Point (goal) and will roll two dice to see whether the player is a loser (rolls a 7) or is a winner (rolls the Point or goal value) or will keep rolling.

It is good practice to create variables with a form scope (field variables) only when necessary. Explain why the variables for the Point (or goal) and state must be form-level (field) variables.

Here is a pseudo-explanation of the program logic for one roll of the dice:

```
Hide all dice images
Roll two dice (get two random numbers)
Display die values, total, and correct images
IF the State is 1 (meaning first roll)
    If total is 7 or total is 11
        status:  Winner
        Add one to the score
        state = 1 (first roll)
    else if total is 2 or total is 3 or total is 12
        status: Loser
        Subtract one from the score
        state = 1 (first roll)
    else
        status: Roll again
        state = 2 (roll again)
        Point = Total the user just rolled
ELSE    (we aren't rolling a first roll, we are rolling again so we need to check the
        roll with the Point - the state must be 2)
    If total is equal to the Point
        status:  Winner
        state = 1 (back to first roll)
        Add one to the score
    else if total 7
        status:  Loser
        state = 1 (back to first roll)
        Subtract one from the score
    else
        status:  Roll again
        state = 2 (roll again)
```

**That should help you understand what needs to be done to create the game.

Test your game. Are you lucky?

Be creative and add your own style to the user interface. Test and debug your program. When everything works, demonstrate your program to your lab instructor. Be sure that you have your lab instructor signs and dates your lab to receive credit.

STEPS FOR SUBMITTING YOUR LAB:

For each lab and following comments must be added at the beginning of your Visual C# code.

/* ‘LAB #

‘SEMESTER NAME

‘STUDENT’S FIRST NAME, LAST NAME

‘I fully understand the following statement.

‘OU PLAGIARISM POLICY

‘All members of the academic community at Oakland are expected to practice and uphold ‘standards of academic integrity and honesty. An instructor is expected to inform and instruct ‘students about the procedures and standards of research and documentation required of students ‘in fulfilling course work. A student is expected to follow such instructions and be sure the rules ‘and procedures are understood in order to avoid inadvertent misrepresentation of his work. ‘Students must assume that individual (unaided) work on exams and lab reports and documentation ‘of sources is expected unless the instructor specifically says that is not necessary.

‘The following definitions are some examples of academic dishonesty:

- ‘Plagiarizing from work of others. Plagiarism is using someone else's work or ideas without ‘giving the other person credit; by doing this, a student is, in effect, claiming credit for ‘someone else's thinking. Whether the student has read or heard the information he/she uses, ‘the student must document the source of information. When dealing with written sources, ‘a clear distinction would be made between quotations (which reproduce information from ‘the source word-for-word within quotation marks) and paraphrases (which digest the ‘source information and produce it in the student's own words). Both direct quotations and ‘paraphrases must be documented. Just because a student rephrases, condenses or selects ‘from another person's work, the ideas are still the other person's, and failure to give ‘credit constitutes misrepresentation of the student's actual work and plagiarism of ‘another's ideas. Naturally, buying a paper and handing it in as one's own work is ‘plagiarism.
- ‘Cheating on lab reports falsifying data or submitting data not based on student's own work.

*/

All labs will be submitted electronically, no paper copies will be given to Lab mentors.

Before submission:

- Please create a folder named as Lab4_FName_LName:
- **Place your solution folder under this folder.**
- **<u>Zip the folder</u>** then upload through Moodle. You will not be able to upload unless you zip, 7zip or rar the folder.

GETTING READY FOR AN INTERVIEW with your Lab Mentor:

The interview is 40% of your lab grade. Make sure to be prepared for your mentor's questions about your program.

When it is your turn to explain your lab to your Lab mentor follow these steps **while your lab mentor is present**:

1. Log on to Moodle.
2. Find your submission link for this lab.
3. Download your Lab on your computer
4. Find your lab wherever you downloaded it to.
5. Make sure to Unzip, (or extract) your folder
6. Open the solution file to demo your lab.

You must follow these steps each time you are being graded for your lab. Your lab mentor must confirm that you downloaded what was submitted on Moodle. You should be graded on what was uploaded on Moodle, not on a local copy obtained from your C drive or external drives (i.e. memory sticks).

HOW WILL YOU BE GRADED BY YOUR LAB MENTOR AND WHAT IS THE GRADING CRITERIA?

1. The application works and was fully tested from what was downloaded and demonstrated from the copy uploaded to Moodle and not from a local copy or any external drive. (50 points)
2. Proper naming conventions were followed as explained in class (10 points)
3. Grade assigned based on oral examination of the students understanding of their solution and the overall quality of the solution (40 points)

GRADE: ________ out of 100