Important Notes

- Deadlines:
  Electronic submission is due on Monday, March 3rd, 2014 11:59pm.
  Problem2 part 1 is due in class on Thursday, February 27th, and
  part 3 is due in class on Tuesday, March 4th, 2014.
- Submission command: submit -c 246 -p **4** -d YourDirectory
- **This assignment is to be done on your own.** If you need help, see the instructor or TA.
- Please start the assignment as soon as possible and get your questions answered early.
- Read through this specification completely before you start.
- Some aspects of this specification are subject to change, in response to issues detected by students or the course staff.

1. **Craps (pp 217-218, 9.8)** (50pts) (due Monday, March 3rd, 2014 11:59pm)
   Write a program that siulates the came of craps, which is played with two dice. One the first roll, the player wins if the sum of the dice is 7 or 11. The player loses if the sum is 2, 3, or 12. Any other roll is called the "point" and the game continues. On each subsequent roll, the player wins if he or she rolls the point again. The player loses by rolling 7. Any other roll is ignored and the game continues. At the end of each game, the program will ask the user whether or not to play again. When the user enters a response other than y or Y, the program will display the number of wins and losses and then terminate.

```
You rolled: 8
Your point is 8
Your rolled: 3
Your rolled: 10
Your rolled: 8
You win!

Play again? y

You rolled: 6
Your point is 6
Your rolled: 5
Your rolled: 12
Your rolled: 3
Your rolled: 7
You lose!

Play again? y
Your rolled: 11
You win!

Play again? n
```

```
Wins: 2   Losses: 1
```

Write your program as three functions: `main`, `roll_dice`, and `play_game`. Here are the prototypes for the latter two functions:

```
int roll_dice(void);
bool play_game(void);
```

Requirements:

- `roll_dice` should generate **two** random numbers, each between 1 and 6, and return their sum.
- `play_game` should play one craps game (calling `roll_dice` to determine the outcome of each dice roll); it will return `true` if the player wins and `false` if the player loses.
- `play_game` is also responsible for displaying messages showing the results of the player's dice rolls.
- `main` will call `play_game` repeatedly, keeping track of the number of wins and losses and displaying the "you win" and "you lose" messages.

Hint: Use the `rand` function to generate random numbers.

2. **Number of days between two dates** (60pts)
Design and implement a program to calculate the number of days between two entered dates. For this problem, a program design is required.

   (a) *Part 1* (due in class on Thursday, February 27th)
   Use Latex to write a program design containing skeleton code and the logic (pseudocode) that represents the implementation of your actual program. All functions are represented by prototypes only, except for main(), which should be fleshed out with appropriate calls and local variable declarations. For this part, submit a hardcopy printout including your pdf file and the Latex source file. No electronic submission required. NO late submission allowed for this part.

   (b) *Part 2* (due Monday, March 3rd, 2014 11:59pm)
   Write a program that calculates the number of days between two entered dates. If the second date is later than the first date, the number of days is negative; if the first date is later than the second date, the number of days is positive.

   Here are rules for deciding a leap year:

   - Every year divisible by 4 is a leap year.
   - However, every year divisible by 100 is not a leap year.
   - However, every year divisible by 400 is a leap year after all.

   Note that days in a month may differ. January, March, May, July, August, October and December each has 31 days, Febuary has 28 days if current year is not a leap year, 29 if it is. All the rest of the months have 30 days.

```
Enter the first date (mm/dd/yyyy):6/30/2006
Enter the second date (mm/dd/yyyy):10/28/2018
Number of days between 6/30/2006 and 10/28/2018 is -4503

Enter the first date (mm/dd/yyyy):10/28/2018
Enter the second date (mm/dd/yyyy):06/30/2006
Number of days between 10/28/2018 and 6/30/2006 is 4503
```

Pay special attention to designing your functions and your program organization. Your project will be graded on its organization and modularity as well as functionality. You should have at least the following functions, but probably more:

- A function that takes a year and decides if it is a leap year or not.
- A function that compares two dates and returns 0 if date1 is same as date 2, a negative number if date1 is earlier than date2, and a positive number if date1 is later than date2.
- A function that computes the number of days between two entered dates.

Error handling: You should NOT assume that users always enter dates correctly. Upon incorrect input, your program either terminates or asks user to enter again. Proper error message should be displayed for user-friendliness.

(c) *Part 3* (due in class on Tuesday, March 4th, 2014)
Use Latex to write how your final implementation differs from your program design, if any, and why. Use the verbatim environment for any code quoted. Submit a printout including your pdf file and the Latex source file in class. No electronic submission required for part 3.