Homework 6

In this experiment, we suppose rand() can generate data with uniform distribution from 0 to RAND_MAX.

Please don’t use self-defined function and array in this lab, but you can use pre-defined functions.

Each problem’s random numbers must be uniform distributed.

1. Write a program to simulate 5,000,000 rolls of six-sided die and output the frequency of each number.
   Let the random seed be 54321 in srand(seed).
   Output to the 7th decimal places for floating points.

   Input/Output Example:

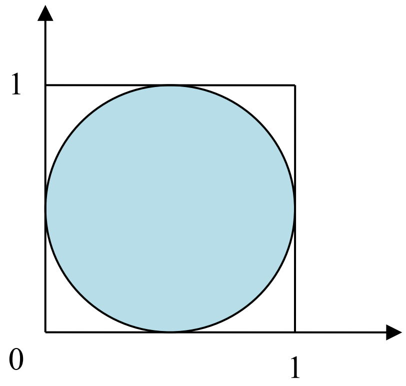
   ```
   1: 0.1668148
   2: 0.1667938
   3: 0.1665812
   4: 0.1665430
   5: 0.1663992
   6: 0.1668680
   ```

2. Charlie tosses a pair of six-sided dice 10,000,000 times. **What number** (sum of the face value of both dice) is **most likely to thrown?** (a 2 is a combination of 1 and 1; a 7 is a combination of 4and 3, 5 and 2, or 6 and 1, and so forth)Please write a program to simulate the process of the toss.
   Let the random seed be 55555 in srand(seed).

   Input/Output Example:

   ```
   Max:   7   0.16683440
   Min:   2   0.02773390
   ```

3. Write a program to simulate throwing darts. (射鏢遊戲)



Use a random number generator to obtain 10,000,000 pairs of floating-point numbers (*x, y*) satisfying $0< x<1$, $0< y<1$, with the following process to generate random number between 0~1:

```
double seed;
const double mpy = 25173.0;
const double inc =13849.0;
const double mod =65535.0;
input variable "seed" then calculate the following formula:
seed = (seed *mpy + inc) % mod; // fmod(seed*mpy + inc, mod)
then, get one random number between 0~1 by using: seed/mod
```

Print the proportion *P* of throws that hit the dart board, that is, the proportion of pairs (*x, y*) that are inside the circle. Also print 4* *P*.

Let user continuously input seed until entering Ctrl+D.
Show the answer to the 7th decimal place.

Notice that the geometry of the problem leads us to expect *P* to be about $\frac{\pi}{4}$. Thus 4* *P* provides an approximation of π.

Hint:

Put the following statements into for loop:

```
     seed = fmod(seed*mpy + inc, mod);
    double x=seed/mod;
    seed = fmod(seed*mpy + inc, mod);
    double y=seed/mod;
```

Input/Output Example:0.123

```
input seed: 0.113
pi/4 = 0.7854723
pi    = 3.1418892

input seed: 0.123
pi/4 = 0.7854681
pi    = 3.1418724

input seed: 0.1225
pi/4 = 0.7853248
pi    = 3.1412992

input seed: ^D
```

4. Newton's method

From Wikipedia, the free encyclopedia You may see more on Wikipedia
This article is about Newton's method for finding roots. For Newton's method for finding minima, see Newton's method in optimization.

In numerical analysis, **Newton's method** (also known as the **Newton–Raphson method**), named after Isaac Newton and Joseph Raphson, is a method for finding successively better approximations to the roots (or zeroes) of a real-valued function.

$$x : f(x) = 0 .$$

The Newton–Raphson method in one variable is implemented as follows:

Given a function *f* defined over the reals *x*, and its derivative *f* ', we begin with a first guess x_0 for a root of the function *f*. Provided the function satisfies all the assumptions made in the derivation of the formula, a better approximation x_1 is

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} .$$

Geometrically, $(x_1, 0)$ is the intersection with the *x*-axis of a line tangent to *f* at $(x_0, f(x_0))$.

The process is repeated as

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Until $|\ x_{n+1} - x_n\ | \leq\ 10^{-8}$

Show the answer to the 8th decimal place and let user continuously input x until entering Ctrl+Z.

Problem:
Please enter one positive integer “data” from the keyboard and find out the square root of this positive integer “data”. (Using Newton’s method only)

Input/Output Example:

```
Input data: 3
sqrt(3) = 1.73205081
Input data: 4
sqrt(4) = 2.00000000
Input data: 5
sqrt(5) = 2.23606798
Input data: 6
```

```
sqrt(6) = 2.44948974
Input data: ^Z
```

5.

A dog is lost in a tunnel at node 0 (see diagram). It can move one node at one time in either direction right or left with equal probability (1 = right, 2 = left).

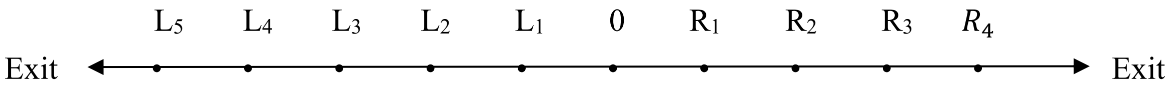
Case a: When the dog hits nodes L2 however, a force of nature always propels(推進) him directly to node L4. The dog escapes from the tunnel when he either hits L5 or R4.

Case b: Let node L2 propel(推進)the dog to L4 only when traveling in a left direction. If node L2 is reached when traveling to the right, the node L2 has no effect.

Start the dog at node 0 10,000,000 times to determine:

(1) Whether the dog has a better chance to exit from the right or the left:
In facts, what are the odds (勝算,可能性) that he will exit from R4? from L5? Show the answer to the 5th decimal place

(2) How long, on the average, the dog stays in the tunnel (each node takes one minute to cover). Show the answer to the 7th decimal place.
Let seed be 55555 in srand(seed).



Input/Output Example:

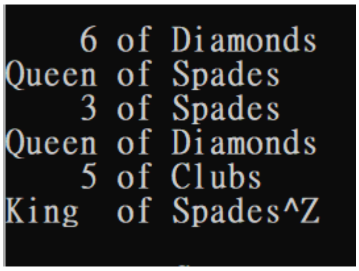
```
seed = 55555, N = 10000000
<== case a ==>
(1)
L5: 66.65773 %
R4: 33.34227 %
The dog has a better chance to exit from L5
(2)
The dog stays in the tunnel for 10.0039831 minutes on the average.

<== case b ==>
(1)
L5: 63.15029 %
R4: 36.84971 %
The dog has a better chance to exit from L5
(2)
The dog stays in the tunnel for 11.5805519 minutes on the average.
```

6. Write a program that displays the name of a card **randomly chosen** from a complete deck of 52 playing cards. Each card consists of a rank (ace, 2,3,4,5,6,7,8,9,10,jack, queen, king) and suit (clubs, diamond, hearts, spades). Your program should display the complete name of the card, as shown in the following sample run: Queen of Spades.

Let the random seed be 54321 in srand(seed).

Let user continuously press “Enter” and stop when pressing Ctrl+Z.

Input/Output Example:



```
    6 of Diamonds
Queen of Spades
    3 of Spades
Queen of Diamonds
    5 of Clubs
King  of Spades^Z
```