Lab 11/08

Please do not use self-defined function, and array in this Lab.
Each problem's random numbers must be uniform distributed.

Lab Part

1. (a) Binomial Distribution

   Toss a coin 8 times, if the result is a head (1), minus 1, else (0) plus 1.
   Sum up the answer 1,000,000 times and calculate the probability of each answer to the 6th decimal place.
   Then visualize the distribution => print '*' (probability/0.01) times.

   (b) Uniform Distributed Fix point numbers

   Please generate 20 random numbers in the range from 0 to 5, the numbers should be accurate to 4 digits after decimal point, and they must be uniformly distributed.
   For example, 1.2345 and 4.8763 each is generated in the same probability.

   (1) Let user continuously input seed until entering Ctrl+Z.
   (2) Set seed once for each input.

Input/Output Example:

```
seed = 12345
-8: 0.003835 *
-6: 0.031304 ****
-4: 0.109342 ***********
-2: 0.219441 **********************
 0: 0.273240 ****************************
 2: 0.219088 **********************
 4: 0.108717 ***********
 6: 0.031110 ****
 8: 0.003923 *
0.0105    2.1466    0.1713    4.2119    1.0046    1.6054    1.2001    2.7248    1.2263    1.9907
1.9032    0.0090    1.2796    0.8481    4.0699    4.9195    4.4376    1.3496    1.7122    2.3346
seed = 54321
-8: 0.003823 *
-6: 0.031396 ****
-4: 0.109325 ***********
-2: 0.218146 **********************
 0: 0.273254 ****************************
 2: 0.219526 **********************
 4: 0.109478 ***********
 6: 0.031202 ****
 8: 0.003850 *
1.8147    4.5513    1.4395    4.1600    2.3045    2.0653    3.3621    0.7502    3.2047    4.1210
2.6241    2.5848    1.1145    3.3249    1.6919    3.9081    0.8722    4.9491    2.8859    0.8464
seed = ^Z
```

2. Simulate n rolls of six-sided die by both methods
   (a) rand() % RANGE + 1
   (b) (int)(rand() / (RAND_MAX + 1.0) * RANGE + 1)

   Then evaluate the KL divergence between the simulation distribution and theoretical distribution of the die.

$$D_{\mathrm{KL}}(P\|Q) = \sum_i P(i) \ln \frac{P(i)}{Q(i)}$$

   There, P is your simulation result and Q is the theoretical value.

   Note: KL divergence is a metric to measure of how one probability distribution P is different from probability distribution Q.

(1) Let user continuously input seed and n until entering Ctrl+Z.
(2) You can use log() in math.h
(3) Both methods should use the same seed.
(4) Please show the answer to the 10th decimal place.

Input/Output Example:

```
seed n: 12345 100
D_KL(a || theory) = 0.0324967288
D_KL(b || theory) = 0.0089461380
seed n: 12345 1000
D_KL(a || theory) = 0.0027379801
D_KL(b || theory) = 0.0009882889
seed n: 12345 10000
D_KL(a || theory) = 0.0002303814
D_KL(b || theory) = 0.0001502950
seed n: 12345 100000
D_KL(a || theory) = 0.0000336025
D_KL(b || theory) = 0.0000290937
seed n: 12345 1000000
D_KL(a || theory) = 0.0000080040
D_KL(b || theory) = 0.0000034485
seed n: 12345 10000000
D_KL(a || theory) = 0.0000001538
D_KL(b || theory) = 0.0000003765
seed n: ^Z
```

3. Understand the following Program and compare the two methods to make the result required by the question.

```
int RANGE = 8000;
srand(9999);
//method (a)
x = rand()%RANGE+1; //generate 1~8000

srand(9999);
//method (b)
int RAND_INV_RANGE=((RAND_MAX+1)/RANGE);
do{
    x=rand();
}while(x >= RANGE * RAND_INV_RANGE);
x/=RAND_INV_RANGE;
x++; //generate 1~8000 */
```

(1)Randomly generate 10^7 times number 1~n positive number by Method (a) and (b), and Count the number of each interval A:[1, L], B:[L+1, 2L], C:[2L+1, 3L], D:[3L+1, 4L] , E:[4L+1, 5L] , F:[5L+1, 6L], where L=n/6. And print the proportion of each interval with percentage.

(2)Let user continuously input seed and n until entering Ctrl+Z.

(3)Input n will be [0, RAND_MAX*(RAND_MAX+2)];

(4)You can use rand()*(RAND_MAX+1)+rand() to generate random number larger than RAND_MAX.

(5)Please show the answer to the 7th decimal place.

Input/Output Example:

```
seed n: 123 100
(a)
A: 0.1599215 B: 0.1700193 C: 0.1701385 D: 0.1598881 E: 0.1699615 F: 0.1700711
(b)
A: 0.1599526 B: 0.1700657 C: 0.1698746 D: 0.1598486 E: 0.1700867 F: 0.1701718

seed n: 123 1000
(a)
A: 0.1660781 B: 0.1669161 C: 0.1668738 D: 0.1659813 E: 0.1671782 F: 0.1669725
(b)
A: 0.1659779 B: 0.1670239 C: 0.1668910 D: 0.1658748 E: 0.1670796 F: 0.1671528

seed n: 123 100000000
(a)
A: 0.1708158 B: 0.1708617 C: 0.1707766 D: 0.1704120 E: 0.1618352 F: 0.1552987
(b)
A: 0.1666372 B: 0.1667033 C: 0.1665846 D: 0.1665021 E: 0.1666993 F: 0.1668735

seed n: ^Z
```

Homework Part

4. 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.

Modification:

(1)5,000,000 rolls -> 10,000,000 rolls

(2)six-sided die -> five-sided die.

Input/Output Example:

```
1: 0.200036
2: 0.200083
3: 0.199921
4: 0.199870
5: 0.200091
```

5. 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 4 and 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).

   Modification:
   (1)A pair of six-sided dice -> three four-sided dice.
   (2)Let the random seed be 55555 in srand(seed) -> Let user continuously input seed and n until entering Ctrl+Z.

   Input/Output Example:

```
seed=12345
Max:  7  0.18750430
Min:  12  0.01568570
seed=54321
Max:  8  0.18734300
Min:  12  0.01559470
seed=^Z
```

6. 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:

….

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 .

Modification:

(1) Circle with radius 0.5 => Ball with radius 0.5 and center (0.5, 0.5, 0.5)

(2) print 4* *P* => 6*P (because $V = \frac{4}{3}\pi r^3$)

(3) Ctrl+D => Ctrl+Z

Input/Output Example:

```
input seed: 0.123
pi/6 = 0.5238083
pi = 3.1428498
input seed: 0.123456
pi/6 = 0.5237181
pi = 3.1423086
input seed: ^Z
```