Question 1

Solution Let us denote the number of girls as $G = 7$ and the number of boys as $B = 9$, and label the circle 1 to $G + B$ counterclockwisely. Define the indicator function $X_i = 1$i f the person at position i and the neighbor in the counterclockwise direction have different genders and $X_i = 0$ otherwise then the number of boy-girl neighbors is given by

$$X = \sum_{i=1}^{G+B} X_i \tag{1}$$

and the expectation is just given by

$$\mathbb{E}(X) = G\mathbb{E}(X_{i\in G}) + B\mathbb{E}(X_{i\in B}) \tag{2}$$

The probability of $X_{i\in G} = 1$ is the position i seats a girl and the position $i + 1$ seats a boy, and by combination

$$\mathbb{P} = \frac{G}{G+B}\frac{B}{G+B-1} \tag{3}$$

therefore

$$\mathbb{E}(X_{i\in G}) = \frac{G}{G+B}\frac{B}{G+B-1} \tag{4}$$

and similarly

$$\mathbb{E}(X_{i\in B}) = \frac{B}{G+B}\frac{G}{G+B-1} \tag{5}$$

Therefore,

$$\mathbb{E}(X) = \frac{GB}{G+B-1} \tag{6}$$

◀

Question 2

Solution Two random variables X and Y are uncorrelated when their correlation coefficient is zero:

$$\rho(X, Y) = 0 \tag{7}$$

and are independent when their joint distribution is the product of their marginal probability distribution,

$$f_{X,Y}(x,y) = f_X(x)f_Y(y) \quad \forall x, y \tag{8}$$

where f is the probability density distribution.

If X and Y are independent, then they are also uncorrelated. However, if X and Y are uncorrelated, they can still be dependent. For example, let X be uniformly distributed on the interval [-1,1], and let Y be

$$Y = \begin{cases} -X & X \leq 0 \\ X & X > 0 \end{cases} \tag{9}$$

Clearly, X, Y has zero correlation coefficient, but they are not dependent. ◀

Question 3

Solution Denote X being the number shown up for a toss. If there is only one toss, the average return is simply

$$E_1 = \frac{1}{6}\sum_{i=1}^{6} i = 3.5 \tag{10}$$

If we have two tosses, we would proceed to the second toss if the first toss is less than the average of the last toss, and otherwise stop. Specifically, we would stop when we get a toss of 4, 5, 6 on the first trial. Therefore, the average return for two tosses will be

$$E_2 = \frac{1}{6}(4+5+6) + \frac{1}{2} \times E_1 = 4.25 \tag{11}$$

Similarly, if there are three tosses, we would not proceed to the second if we get a 5 or 6 on the first trial. Subsequently, the expectation of three tosses will be

$$E_3 = \frac{1}{6}(5+6) + \frac{4}{6} \times E_2 = \frac{14}{3} \tag{12}$$

◀

Question 4

Question 5

Solution Correlation is semidefinite, and therefore we have

$$A = \begin{pmatrix} 1 & r & 0 \\ r & 1 & r \\ 0 & r & 1 \end{pmatrix} \tag{13}$$

has to have all positive eigenvalues. The eigenvalue of this matrix is

$$\lambda_1 = 1 \tag{14}$$

$$\lambda_2 = 1 + \sqrt{2}r \tag{15}$$

$$\lambda_2 = 1 - \sqrt{2}r \tag{16}$$

The positive semidefinite condition is equivalent to have nonnegative eigenvalues, which set the boundary of r as

$$-\frac{\sqrt{2}}{2} \leq r \leq \frac{\sqrt{2}}{2} \tag{17}$$

◀

Question 6

Solution Let $X_i = 1$ if the drunk man takes a step left at time i and $X_i = -1$ if the drunk man takes a step right. Take the left door as $X = 0$, then the position of the drunk man at time n is

$$S_n = \sum_{i=1}^{n} X_i \tag{18}$$

with

$$S_0 = 1 \tag{19}$$

when the left door is not locked, X_i is a random walk, and

$$S_n, \quad S_n^2 - n \tag{20}$$

are obviously martingales. In addition, the time N the drunk man gets out of the door is a stop time, and a martingale at stopped time is also a martingale, i.e.

$$S_N, \quad S_N^2 - N \tag{21}$$

are still martingales. Denote the probability of getting out from the left door is p, and then the probability of getting out from the right door is $1 - p$. Therefore

$$\mathbb{E}[S_N] = p \times 0 + (1 - p) \times 100 = 1 \tag{22}$$
$$\mathbb{E}[S_N^2 - N] = p \times 0 + (1 - p) \times 100^2 - \mathbb{E}[N] = S_0^1 - 0 = 1 \tag{23}$$

solving this, yielding

$$p = \frac{99}{100} \tag{24}$$
$$\mathbb{E}[N] = 99 \tag{25}$$

◀

Question 7

Solution ◀

Question 8

Solution ◀

Question 9

Solution A simple expectation of this game is given by

$$2 \times \frac{1}{2} + 4 \times \frac{1}{4} + 8 \times \frac{1}{8} + \cdots = \infty \tag{26}$$

However, we really need to maximum utility rather than the expected return from an economic view because does not feel too much different to have

another \$1 billion when they already have \$1 billion. In other words, the sum has to have a cutoff in some sense, and let this to be \$1 billion, or roughly 2^{30}, we have

$$2 \times \frac{1}{2} + 4 \times \frac{1}{4} + 8 \times \frac{1}{8} + \cdots + 2^{30} \times \left(\frac{1}{2^{30}} + \frac{1}{2^{31}} + \cdots \right) = 31 \qquad (27)$$

which is not infinite any more. ◀

4. If a rod with length L is cut randomly into N pieces, what is the distribution of the longest piece?

Solution Without loss of generality, suppose $L = 1$. Let $X_1, X_2, \cdots, X_{n-1}$ denote the ordered distances of cut positions from the origin. Let X_0 and X_n denote the beginning and the end of the rod, respectively, such that $X_0 = 0$ and $X_n = 1$. Let $V_i = X_i - X_{i-1}$, then the V_i's are the lengths of the pieces of rod. Note that the probability that any particular k of the V_i's simultaneously have lengths longer than $c_1, c_2, \cdots, c_k$, respectively (where $\sum_{i=1}^{k} c_i \leq 1$), is

$$\mathsf{Pr}(V_1 > c_1, V_2 > c_2, \cdots, V_k > c_k) = (1 - c_1 - c_2 - \cdots - c_k)^{n-1} \tag{1}$$

This is proved formally in David and Nagaraja's *Order Statistics*, p. 135 (see below). Intuitively, the idea is that in order to have pieces of size at least $c_1, c_2, \cdots, c_k$, all $n-1$ of the cuts have to occur in intervals of the rod of total length $1 - c_1 - c_2 - \cdots - c_k$. To see this, we can look at the range of each single point by drawing a graph. Take the first cut position X_1 for example. X_1 must be located in a range such that $V_1 > c_1$ and $1 - V_1 > c_2 + c_3 + \cdots + c_k$ (see Figure 1 for illustration). Hence, the range of X_1 (enclosed by square brackets in Figure 1) is $1 - c_1 - c_2 - \cdots - c_k$. Similarly, the range of other cut positions $X_2, X_3, \cdots, X_{n-1}$ is also $1 - c_1 - c_2 - \cdots - c_k$. Together, these facts culminate in equation (1).

Here is a formal proof of equation (1):

Denote the positions on the rod where the cuts are made (unordered) by $U_i (i = 1, \cdots, n-1)$. Since all cuts are random, we have $U_i \overset{iid}{\sim} \mathsf{Unif}(0, 1)$, for $i = 1, 2, \cdots, n-1$. Then we have the joint pdf of order statistics $X_1, X_2, \cdots, X_{n-1}$ over the simplex $0 \leq x_1 \leq \cdots \leq x_{n-1} \leq 1$:

$$\begin{aligned}
& f(X_1 = x_1, X_2 = x_2, \cdots, X_{n-1} = x_{n-1}) && (2)\\
&= (n-1)! f(U_1 = x_1, U_2 = x_2, \cdots, U_{n-1} = x_{n-1}) && (3)\\
&= (n-1)! f(U_1 = x_1) f(U_2 = x_2) \cdots f(U_{n-1} = x_{n-1}) && (4)\\
&= (n-1)! && (5)
\end{aligned}$$

Since $V_i = X_i - X_{i-1}$, the determinant of the Jacobian matrix J that maps

from X_i's to V_i's is:

$$\det J = \begin{vmatrix} \frac{\partial V_1}{\partial X_1} & \cdots & \frac{\partial V_1}{\partial X_{n-1}} \\ \vdots & \ddots & \vdots \\ \frac{\partial V_{n-1}}{\partial X_1} & \cdots & \frac{\partial V_{n-1}}{\partial X_{n-1}} \end{vmatrix} \tag{6}$$

$$= \begin{vmatrix} 1 & 0 & 0 & \cdots & 0 \\ -1 & 1 & 0 & \cdots & 0 \\ 0 & -1 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{vmatrix} \tag{7}$$

$$= 1 \tag{8}$$

By change of variable, the joint pdf of V_i's is

$$f(V_i = v_1, V_2 = v_2, \cdots, V_{n-1} = v_{n-1}) = \frac{f(X_1 = x_1, X_2 = x_2, \cdots, X_{n-1} = x_{n-1})}{\det J} \tag{9}$$

$$= (n-1)! \tag{10}$$

where $v_i \geq 0$ and $\sum_{i=1}^{n-1} v_i \leq 1$. It follows that the joint pdf of $V_1, \cdots, V_k$ is, for $\sum_{i=1}^{k} v_i \leq 1$,

$$f(V_1 = v_1, \cdots, V_k = v_k) = (n-1)! \int_0^{1-v_1-\cdots-v_k} \cdots \int_0^{1-v_1-\cdots-v_{n-2}} \mathrm{d}v_{n-1} \cdots \mathrm{d}v_{k+1} \tag{11}$$

Note

$$\int_0^{1-v_1-\cdots-v_k} \cdots \int_0^{1-v_1-\cdots-v_{n-2}} \mathrm{d}v_{n-1} \cdots \mathrm{d}v_{k+1} \tag{12}$$

$$= \int_0^{1-v_1-\cdots-v_k} \cdots \int_0^{1-v_1-\cdots-v_{n-3}} (1 - v_1 - \cdots - v_{n-2}) \mathrm{d}v_{n-2} \cdots \mathrm{d}v_{k+1} \tag{13}$$

Set $z = 1 - v_1 - \cdots - v_{n-2}$. Then $\mathrm{d}v_{n-2} = -\mathrm{d}z$. When $v_{n-2} = 0$, $z = 1 - v_1 - \cdots - v_{n-3}$; when $v_{n-2} = 1 - v_1 - \cdots - v_{n-3}$, $z = 0$. Hence, by change

of variable,

$$(14) = \int_0^{1-v_1-\cdots-v_k} \cdots \int_{1-v_1-\cdots-v_{n-3}}^{0} (-z)\mathrm{d}z \cdots \mathrm{d}v_{k+1} \tag{14}$$

$$= \int_0^{1-v_1-\cdots-v_k} \cdots \int_0^{1-v_1-\cdots-v_{n-4}} \frac{(1-v_1-\cdots-v_{n-3})^2}{2} \mathrm{d}v_{n-3} \cdots \mathrm{d}v_{k+1} \tag{15}$$

$$= \cdots = \frac{(1-v_1-\cdots-v_k)^{n-k-1}}{(n-k-1)!} \tag{16}$$

Therefore,

$$(11) = \frac{(n-1)!}{(n-k-1)!}(1-v_1-\cdots-v_k)^{n-k-1} \tag{17}$$

For constants $c_i \geq 0$ $(i = 1, \cdots, k)$ with $\sum_{i=1}^{k} c_i \leq 1$, we have

$$\mathsf{Pr}(V_1 > c_1, \cdots, V_k > c_k) \tag{18}$$

$$= \int_{c_1}^{1} \int_{c_2}^{1-v_1} \cdots \int_{c_k}^{1-v_1-\cdots-v_{k-1}} f(v_1, \cdots, v_k)\mathrm{d}v_k \cdots \mathrm{d}v_1 \tag{19}$$

$$= (1-c_1-\cdots-c_k)^{n-1} \tag{20}$$

Let $V_{(n)}$ denote the longest piece of rod, then

$$\mathsf{Pr}(V_{(n)} > x) = \mathsf{Pr}(V_1 > x \text{ or } V_2 > x \text{ or } \cdots \text{ or } V_n > x) \tag{21}$$

According to the principle of inclusion/exclusion, namely

$$\mathsf{Pr}\left(\bigcup_{i=1}^{n} \mathrm{A}_i\right) = \sum_{i=1}^{n} \mathsf{Pr}(\mathrm{A}_i) - \sum_{i<j} \mathsf{Pr}(\mathrm{A}_i \cap \mathrm{A}_j) + \sum_{i<j<k} \mathsf{Pr}(\mathrm{A}_i \cap \mathrm{A}_j \cap \mathrm{A}_k) - \cdots + (-1)^{n-1} \mathsf{Pr}\left(\bigcap_{i=1}^{n} \mathrm{A}_i\right) \tag{22}$$

we then have the distribution of $V_{(n)}$ as follows:

$$\mathsf{Pr}(V_{(n)} > x) = n(1-x)^{n-1} - \binom{n}{2}(1-2x)^{n-1} + \cdots + (-1)^{k-1}\binom{n}{k}(1-kx)^{n-1} + \cdots \tag{23}$$

where $0 < x < 1$ and the sum continues until $kx > 1$. ◀

1 Reference

https://books.google.com/books?id=bdhzFXg6xFkC&pg=PA133&lpg=PA133&
dq=david+and+nagaraja+order+statistics+division+of+random+interval&
source=bl&ots=OaK_l7ycZk&sig=BADI1rALE7hGm3-tvR6du1XKKzk&hl=en&ei=
G68GTcqPEIeusAOZmdC4Bw&sa=X&oi=book_result&ct=result#v=onepage&q=
david%20and%20nagaraja%20order%20statistics%20division%20of%20random%
20interval&f=false

http://math.stackexchange.com/questions/14190/average-length-of-the-longest-seg
lq=1

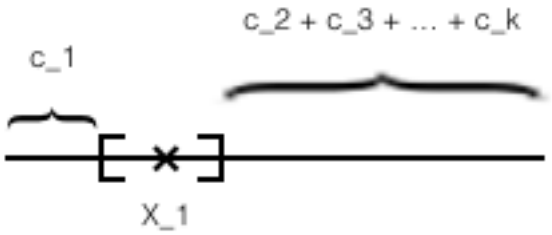

Figure 1: Range of the first cut position on the rod

- Question 10:

  Solution:

  Reference: http://www.cplusplus.com/doc/tutorial/classes2/

  The default constructor is the constructor called when objects of a class are declared, but are not initialized with any arguments. If a class definition does not have constructors, the compiler will assume that the class have an implicitly defined default constructor. However, as soon as a class has constructor taking parameters explicitly declared, the compiler will no longer provide an implicit default constructor, and will no longer allow the declaration of new objects of that class without arguments. In order to declare a new object that does not take any parameters, at the presence of some constructors with arguments, we need to explicitly define our own default constructor.

- Question 11:

  Solution:

  A singleton pattern is a design pattern that ensures a class has only one instance, and provides a global point of access to it.
  The general method to make a program thread safe is to lock shared resources whenever write permission is given. This way, if one thread is modifying the resource, other threads can modify it.

- Question 12:

  Solution:

```cpp
// C++
// Dynamic Programming

class Solution {
public:
    int maxProfit(vector<int> input, int k) {
        if(input.size() < 2) {
            return 0;
        }
        int res = 0;
        if (k >= input.size()){
            for (int i = 1; i < input.size(); ++i) {
                if (input[i] - input[i - 1] > 0) {
                    res += input[i] - input[i - 1];
                }
            }
        } else {
            vector<int> local(k + 1);
            vector<int> global(k + 1);
            for (int i = 0; i < input.size() - 1; ++i) {
                int increase = input[i + 1] - input[i];
                for (int j = k; j >= 1; --j) {
                    local[j] = max(global[j - 1] + max(increase, 0),
```

```
                    local[j] + increase);
                    global[j] = max(global[j], local[j]);
                }
            }
            res = global[k];
        }
        return res;
    }
};
```

- Question 13:

  Solution:

```
// C++
// XOR

class Solution {
public:
    int findSingle(vector<int> input) {
        // assume the vector is not null
        int x = input[0];
        for (int i = 1; i < input.size(); ++i) {
            x = x^input[i];
        }
        return x;
    }

    void test1() {
        int a[] = {5,3,2,4,3,4,2};
        vector<int> v_a(a, a + sizeof(a)/sizeof(int));
        cout << findSingle(v_a) << endl;
    }
};
```

  This is a variant of what is often called "the factory pattern".
  The point is that user() is completely isolated from knowledge of classes such as AX and AY.

- Question 14:

  Solution:

  Reference: https://isocpp.org/wiki/faq/strange-inheritance#calling-virtuals-from-base

  Yes. It is allowed. If all derived classes have the same algorithm for method A. Method A needs to call method B. However, each derived class has their own algorithm to implement method B. In this case, method A could be non-virtual while method B should be virtual function.

- Question 15:

Solution:

Reference: http://pit-claudel.fr/clement/blog/how-random-is-pseudo-random-testing-pseudo-random-number-generators-and-measuring-randomness/
http://www.drdobbs.com/testing-random-number-generators/184403185

True random number generators rely on a physical source of randomness: thermal noise, cosmic noise, nuclear decay etc.
Pseudo-random number generator takes one number(seed) and produce a sequence of bits. There are various ways to test the randomness: equidistribution, binary matrix rank, discrete fourier transform, compressibility, maximum distance to zero, average flight time, random excursions, etc.

- Question 16:

  Solution:

```
// C++
// coin toss will produce random number (0,1)
// three coin tosses will produce random numbers 000 - 111
// remove 000 and 111 if they appear, keep 001 - 110, or 1 - 6
class Solution {
private:
    int coin() {
        return rand()%2;
    }

public:
    int dice() {
        int one;
        int two;
        int three;
        while(1){
            one = coin();
            two = coin();
            three = coin();
            int sum = one + two + three;
            if (sum != 0 && sum != 3){
                return 4*one + 2*two + three;
            }
        }
    }
};
```

- Question 17:

  Solution:

```
// C++

void MoveRobot2Meet()
{
        unsigned int i=0;
        unsigned int backup_i;
```

```
        while(1)
        {
                backup_i = i;

                  //Check left
                while(i)
                {
                        GoLeft();
                        --i;

                }
                if(at-zero())
                      return;

                i = ++backup_i;

                 //Check right
                while(i)
                {
                        GoRight();
                        --i;
                }

                if(at-zero())
                     return;

                i = ++backup_i;
        }
}
```