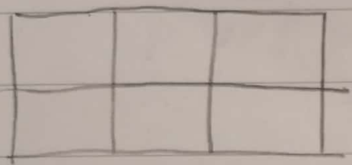
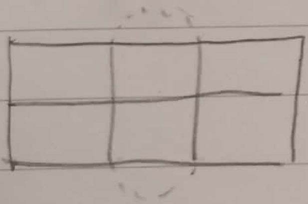
2. It is a non-Eulerian graph. We neet to use mimimum cost to turn it to a Eulerian graph

A graph is Eulerian if two vertices have odd degree and all vertices in a connected graph



there are 6 vertices with odd degree.

We can add 2 edges to eliminate 4 vertices with odd degree, thus the new graph is a Eulerian trail.



Thus, the shortest route is 17+2=19 long

1. Chi-square text is used to text the chance that the difference between two categorical dataset is generated by randomness. It can test independence & goodness of fit. Yates' X^2-text is used to text independence in a contingency table

3. $P(X \mid X+Y>0)$

$$= \frac{P(X, X+Y>0)}{P(X+Y>0)}$$

$$= \frac{P(X+Y>0 \mid X)\,P(X)}{P(X+Y>0)}$$

X, Y are i.i.d. $N(0,1)$

so X+Y is gaussian too

$$= \frac{P(Y>-X \mid X)\,P(X)}{P(X+Y>0)}$$

$$= \frac{(1-P(X))\,P(X)}{\frac{1}{2}}$$

$$= -2P^2(X) + 2P(X)$$

done

4. 1. $E(\text{loss}) = \frac{1}{6} \cdot 35 = \frac{35}{6}$

$E(\text{gain for next rounds}) = \frac{1}{6}(2+3+4+5+6) + \frac{1}{6}(1+2+3+4+5+6) = \frac{41}{6}$

$E(\text{gain}) > E(\text{loss})$, should continue

2, <u>44</u> is most probable amount.

$43-35=8$ if next dice rolls as 1, then stop

Thus possible path are:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1 roll	*	1	1	1	1	1	0	0						
2 rolls		0	1	2	3	4	5	4	3	2	1	0	0	0
3 rolls				1	3	6	10	15	19	21	21	19	15	10
4 rolls					1	4	10	20	35	54	74	92	105	110
5 rolls						1	5	15	35	70	124	149	285	380
6 rolls							1	6	21	56	126	250	446	726
7								1	7	28	84	210	460	905
8									1	8	36	122	332	792
9										1	44	166	418	1290

5

$$E \cdot L_{small} = \int_0^{\frac{1}{3}} \int_x^{1-2x} \frac{1}{1-x}\, dy\, x \cdot dx$$

$$= \int_0^{\frac{1}{3}} \frac{1-3x}{1-x} \cdot x\, dx$$

$$= \int_0^{\frac{1}{3}} \left(3x - \frac{2x}{1-x}\right) dx$$

$$= \frac{3x^2}{2}\Big|_0^{\frac{1}{3}} + 2\Big|_0^{\frac{1}{3}} - \int_0^{\frac{1}{3}} \frac{2}{1-x}\, dx$$

$$= \frac{1}{6} + \frac{2}{3} + 2\log\frac{2}{3}$$

x

$1-2x$

$1-x$

$$E\, L_{big} = \int_{\frac{1}{3}}^{\frac{1}{2}} \left(\frac{1}{1-x}\int_{1-2x}^{x} dy\right) x\, dx$$

$$= \int_{\frac{1}{3}}^{\frac{1}{2}} \left(\frac{3x-1}{1-x}\right) x\, dx$$

$$= \int_{\frac{1}{3}}^{\frac{1}{2}} \left(-3x + \frac{2x}{1-x}\right) dx$$

$$= -\frac{3x^2}{2}\Big|_{\frac{1}{3}}^{\frac{1}{2}} - 2x\Big|_{\frac{1}{3}}^{\frac{1}{2}} - 2\log(1-x)\Big|_{\frac{1}{3}}^{\frac{1}{2}}$$

$$= \frac{1}{6} - \frac{3}{8} - \frac{1}{3} - 2\left(\log\frac{1}{2} - \log\frac{2}{3}\right)$$

$$E\, L_{middle} = 1 - EL_{small} - EL_{big}$$

6 total # of combinations is $N!$

(1)(2) # of k people have wrong hats:

$$\binom{k}{N}\, !k \quad \text{where } !k = k! \sum_{i=0}^{k} \frac{(-1)^i}{i!}$$

$$\therefore E(k) = \sum_{k=0}^{N} \frac{1}{N!} \binom{k}{N}\, !k \cdot k$$

$$= \sum_{k=0}^{N} \frac{1}{(N-k)!} \left(\sum_{i=0}^{k} \frac{(-1)^i}{i!} \right) \cdot k$$

$$E(k^2) = \sum_{k=0}^{N} \frac{1}{N!} \binom{k}{N}\, !k \cdot k^2$$

$$= \sum_{k=0}^{N} \frac{1}{(N-k)!} \left(\sum_{i=0}^{k} \frac{(-1)^i}{i!} \right) k^2$$

$$Var(k) = E(k^2) - (Ek)^2$$

(3)

Prepared by | Index no.

Date

$\sum(\hat{y}_i' - \bar{y}_i)^2 = 0.1 \cdot SSTO$, $\sum(\hat{y}_i^2 - \bar{y}_i)^2 = 0.2 \cdot SSTO$

7. $\sum(y_i - \hat{y}_i)^2 = 0.9 \cdot SSTO = \sum(y_i - \beta_1 x_{1,i})^2 = 0.9 \cdot SSTO$

$\sum(y_i - \hat{y}_i)^2 = 0.8 \cdot SSTO = \sum(y_i - \beta_2 x_{2,i})^2 = 0.8 \cdot SSTO$

$\sum(\bar{y}_i - \alpha_1 x_i^1 - \alpha_2 x_i^2)^2$

$= \sum(\bar{y}_i - \alpha x_i^1)^2 - \sum 2(\bar{y}_i - \alpha_1 x_i^1) \cdot \alpha_2 x_i^2 + \sum(\alpha_2 x_i^2)^2$

$\leq 0.1 \cdot SSTO + \sum(-2\bar{y}_i \alpha_2 x_i + (\alpha_2 x_i^2)^2) + \sum 2\alpha_1\alpha_2 x_i^1 x_i^2$

$= 0.1 \cdot SSTO + \sum(\alpha_2 x_i - \bar{y}_i)^2 - \sum(\bar{y}_i)^2 + \sum 2\alpha_1\alpha_2 x_i^1 x_i^2$

$\leq 0.1 \cdot SSTO + 0.2 \cdot SSTO + \sum 2\alpha_1\alpha_2 x_i^1 x_i^2 - \sum(\bar{y}_i)^2$

$\leq 0.3\ SSTO$

$\therefore$ r-squared in $[0.2, 0.3]$

Prepared by | Index no.

Date

8. assume $2^k < n < 2^{k+1}$

Then any path with begining of T is considered as failure

a path consisting of all (k+1) H is assumed as accept

Choose $(n-1-2^k)$ randomly from all other paths starting with H as failures.

The rest are regarded as retry.

9. $E(L) = \frac{1}{2}(10) + \frac{1}{2}\cdot\frac{1}{2}(1+10) + \frac{1}{2^3}(2+10)$

$+ \cdots \frac{1}{2^{10}}(9+10) + \frac{1}{2^{10}}\cdot 10$

$= 10 + \underbrace{\frac{1}{4} + \frac{2}{8} + \cdots \frac{9}{2^{10}}}_{A}$

$A - \frac{1}{2}A = \frac{1}{4} + \frac{1}{8} + \frac{1}{2^{10}} - \frac{9}{2^{11}}$

$\frac{1}{2}A = \frac{\frac{1}{4}(1-(\frac{1}{2})^9)}{1-\frac{1}{2}} - \frac{9}{2^{11}}$

$= 10 + 1 - \frac{11}{2^{10}}$

$= 11 - \frac{11}{2^{10}}$

11. class MyNode:
    def __init__(self, val)
        self.value = val
        self.next = None

To delete the node next to A, defined as temp

temp = A.next ; A.next = temp.next.

13.
1. No, to create an object, complete information is needed.

2. similar function as virtue constructor

Base * p = new B( )

Base * P1 = new p-> A( )

where B(.) is a derived class of A(..)

14 Yes, it is.

Prepared by | Index no.

Date

15. will implement it using recursive method:

```
def myfun (mystr, i, j):
    if i = j:
        return 1
    if j = i+1 & mystr(i] = mystr[j]
        return 2
    if mystr[i] = mystr [j] :
        return myfun [mystr, i+1, j-1] +2
    if mystr(i] != mystr[j] :
        retun max (myfun (seq, i, j-1), myfun(seq, i+1, j))
```

16. First, split the string by '␣'. it generates a list.

Second, traverse the list.

Third: '␣'.join(List) to form a new string.

Prepared by

Index no.

17. psudo-code:

1. find the local minima & day 0 is local minima if $P_0 < P_1$: [min_1, min_2, ...]
2. find the local maxima & day N is local maxima if $P_N > P_{N-1}$, [max_1, max_2, ...]

3. choose two largest number from [$max_1 - min_1$, ... $max_k - m_k$]

18. design a recursive solution:

or condition 'i' is not connected in the residual

```
def myfun (N, List , index]
    if i not in index & len(index)>0 ;return -1
    for i in List[0]:
        N1 = N-1
        List1 = List.remove(list(List(0] [i]))
        index1 = index.remove(i)
        result.append[i]
    val = myfun(N1, List1, index1)
        if val ==-1:
            result.remove[i] ; return -1
    else:
        return i
```