1. By Duyen Tran

X: # of customers that get their own nat back

X: Indicator Random Variable associated with the event that customer i gets their own hat back

Xi= I { customer i gets their hat back } = { 0 if they do it they don't [[xi] = fr(xi gets their hat back)

$$E[X] = E\left[\sum_{i=1}^{N} X_i\right]$$

$$= \sum_{i=1}^{N} E[X_i]$$

$$= \sum_{i=1}^{N} \frac{1}{N}$$

$$= \sum_{i=1}^{N} \frac{1}{N}$$

$$= \sum_{i=1}^{N} \frac{1}{N}$$

the expected # of customers who will get their own hat back is 1

2. By James Tan

Indicator Random Variable: The variable that A[i] > A[j]; 1 if
True OF not CACI Same the reconstation is uniform random, we
know nothing about the across except that all the number are distinct,
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1/2 For False
$F[x] = F \left[\begin{array}{c} x \\ \end{array} \right]$
change of variable holding true
In that case, the probability of the variable string time is to false. E[x] = E S X: Change of variable hilling true Cinbratur contain variable)
From 1 to n-1 In array because comparison of
last element connot next element to figh
pe embreal 10, 17511 man to U
$= \sum_{i=1}^{n-1} \sum_{i=1}^{n} E[X_{ij}] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \sum_{j=i+1}^{n-1} \cdot \left(\frac{1}{2}(n-(i+1)+1)\right)$
$\sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{j=1}^{n} \sum_{j=1}^{n} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{j$
$= \frac{1}{2} \sum_{i=1}^{n-1} (n-i) = \frac{1}{2} \sum_{i=1}^{n-1} n - \frac{1}{2} \sum_{i=1}^{n-1} 1 = \frac{1}{2} [n \cdot (n-1)] - \frac{1}{2} [\frac{1}{2} (n-1)(n)]$
1 +2+ +n-2+n-1
n-1+n-2++2+1
Y
2 = (n-1) n
$\Sigma = \frac{(n-1)(n)}{2}$
$= \frac{1}{2} \left[n^2 - n \right] - \frac{1}{2} \left[\frac{1}{2} \left(n^2 - n \right) \right] = \frac{1}{2} \left[n^2 - n - \frac{1}{2} n^2 + \frac{1}{2} n \right]$
$=\frac{1}{2}\left[\frac{1}{2}n^2-\frac{1}{2}n\right] = \frac{1}{4}\left[n^2-n\right] = \frac{n \cdot (n-1)}{4}$
= 2 [20 - 2"] - 4 [" "] - 4

3. By Bonnie Liu

PERMUTE-WITH-ALL(A)

- $1 \quad n = A.length$
- 2 **for** i = 1 **to** n
- swap A[i] with A[RANDOM(1, n)]

Does this code produce a uniform random permutation? Why or why not?

Let n=3, then there are 27 possible outcomes of calling permute - nill-all Assume permute-nill-all produce a uniform random permutation, then $3!=3k2\times 1=6$, for each permutation would occur times.

So each permutation have to occur i times. but no integer i can satisfies $\frac{1}{27} = \frac{1}{6}$.

from above we know this coole does not produce a uniform rondom permutation.

	1. a. Partition will return ger so, with all elements
	In the array former and the same
	T(n) = T(n) 1 2000 / me recurrence will be
	In the array being equal, the recurrence will be $T(n) = T(h-1) + \Theta(n) + T(c)$ for the running time.
	(6) 20(00)
	b. Partition (A,p,r)
	$1 \times = A(p)$
	$2 \qquad \bar{l} = h = \rho$
	3 for 7=0+1 to r
	3 for 5= p+1 to r 4 if ACi] < x
-	$ \begin{array}{ccc} S & y = A C_{2} \\ C & A(J) = A (h+1) \end{array} $
-	$C \qquad A(J) = A(h+1)$
-	$7 \qquad \qquad A(b+1) = A(c)$
	ACJ = y
	9 = 3 + 10
	b = b + 1
	N else if AG] = = x
	12 exchange A(h+1) was A(s)
	12 exchange AChti) with ACi) 13 h=h+1
	14 return (ich)
	C R. James and a R. Jahre CA
13	C. Randomszed-Partition (A, p, r)
13-	(= Kardom (pr)
and the second second second	2 exchange ACJ with ACE]
	2 exchange ACT with ACET 3 return Partition (A.p.r)
	Quick sort (A,p,r)
	1 if 950
	2 (q,i) = Randomized-Partition (A, q,r) 3 Quicksort (A, p, q-1) 4 Quicksort (A, i+1, r)
	3 Quicksort (A. Q. a-1)
adipanses islant from	4 Buschsort (A, 1+1, C)
disabunish (ilintrovindo)	
approximation are from	d An almost & the time sound to the orest should be get in
AND THE PERSON NAMED IN	at my premient of that the egod to me protection of the received
to simulation to sign and the	the same partition as pivot the don't live
overometricum draw Mile	d. Any elements that're equal to the pivot should be put in the same partition as pivot because we don't recurse on those elements. So the subgration size with Quicksort isn't larger than the subgration size of Quicksort even with equal elements.
	Quicksort isn't larger than the subproblem size
	of auschood even with equal elements,