Poker Sort

Hand = 5 cards

1. xStraight Flush
    1. Flush - All cards from same suit
    2. Straight - 5 cards in ascending order
    3. Ace can be 0 or 14
        1. Ex. A 2 3 4 5 and 10 J Q K A
2. four-of-a-kind
    1. If there are 2 four-of-a-kind the one with the highest card number trumps
3. full house
    1. (3-of-a-kind and a pair)
4. Flush
    1. all same suit - lexicographic order
5. 3-of-a-kind
    1. 2 pairs
6. 1 pair
7. high card

Hint: Sort hand

Card Index to be used in Code	Actual Card Number/Letter
0	2
1	3
2	4
3	5
4	6
5	7
6	8
7	9
8	10
9	j
10	q
11	k
12	a

Clubs	Diamonds	Hearts	Spades
0	1	2	3

$0 \leq (\text{card} = 4 \cdot \text{rank} + \text{suit}) \leq 51$

```
hand = vector<int>;
//----------------------
void sort(vector<Hand>& hands)
```

Ex.

K_h = 46

7_s = 23 ...

Probability

Average Runtime

Sample Space : Ω

Ω is a set, $\omega \in \Omega$

Ex: Ω = set of all poker hands $\omega = C_1, C_2, \ldots C_5$

(Discrete) Assign number, $0 \leq p \leq 1$ to each $\omega \in \Omega$ satisfy probability of a set is sum of probabilities of elements.

$P(a) = \sum_{\omega=A} p(\omega)$

Require: $P(\Omega) = 1$

Aside

$p(\omega)$ represents degree of belief.

AN event is a subset $A \subset \Omega$

The "smallest" events {ω} are atomic events

(Aside: In infinite Ω can have unmeasurable events...)

A discrete prob distribution is $p : \Omega \to [0, 1]$

More generally, a prob density $p : \Omega \to [0, 1]$ and $P(A) = \int_A P(t)dt$

Uniform Distribution of [0,1], p(t) = 1

$p(\frac{1}{3}, \frac{2}{3}) = \int_{\frac{1}{3}, \frac{2}{3}} 1dt = \frac{1}{3}$

Expected Value: $E(A) = \int_A tp(t)dt$

11/1/2021 Probability

Ω = sample space

$A \subset \Omega$ an event

P: Power set of Ω -> [0,1]

If A n B = 0 then p(A u B) = P(A) + P(B)

Do Now

```
bool bad_flip(){
    //returns true with probability p but p is unknown.
}
//Using bad_flip(), write fair_flip() that returns true with probability 1/2
bool fair_flip(){
    while(true){
        bool a = bad_flip(), b = bad_flip();
        if(a^b){
            return a;
        }
    }
    //probability 1 is p(1-p) and probability 2 is (1-p)p
}
```

Conditional Probability

P(A | B) - Probability of A given B

"A given B"

P(A | B) = P(A n B)/

$p(A|B) = \frac{P(A \cap B)}{P(B)}$

Independent Events

A, B are independent if P(A | B) = P(A)

Note: if P(A | B) = P(A), we have that $\frac{P(A \cap B)}{P(B)} = P(A) \rightarrow P(A \cap B) = P(A)P(B)$

Events $A_1, \ldots, A_n$ are pairwise independent if $P(A_i \cap A_j) = P(A_i)P(A_j) \forall i \neq j$

Events $A_1, \ldots, A_n$ are independent if
$P(A_1, \ldots, A_{i_k}) = P(A_i) \cdot \ldots \cdot P(A_{i_k}) \forall k \& subsets \{A_1, \ldots, A_{i_k}\} \subset \{1, 2, 3 \ldots, n\}$

Ex. Flip 2 coins

$A_1 =$ 1st heads $P(A_1) = \frac{1}{2}$

$A_2 =$ 2nd heads $P(A_2) = \frac{1}{2}$

$A_3 =$ coins are different. $P(A_3) = \frac{1}{2}$

$P(A_1 A_2) = \frac{1}{4}$

$P(A_1 A_3) = \frac{1}{4}$

$P(A_2 A_3) = \frac{1}{4}$

$P(A_1 A_2 A_3) \neq \frac{1}{8}$

$P(A_1 A_2 A_3) = 0$

Game of Craps

Roll 2 dice.

7 or 11 --> win

2, 3, 12 --> lose

else(rolled 4,5,6,8,9,10){

let point = value rolled

keep rolling until

7 -> lose

point -> win

}

Calculate the possibility of winning (assume fair dice)

11/2/2021 Distribution and Density

X is a random

Ex. Bernoulli random variable:

Sample space Ω

For example if this was a simple coin toss problem

$\Omega = \{heads, tails\}$

X(heads) = 1

X(tails) = 0

$P(x \leq a) = P(\omega \in \Omega : X(\omega) < a)$

We define $F : R \to [0, 1]$ by $F(t) = P(X < t)$

This is a cumulative Distribution function (CDF)

Lets say heads has a probability p and tails has a probability of 1-p

$F(t) = 1 - p$ if t<1 and $F(t) = p$ if t=1

Any monotonous increasing function that is right-continusous $0 \leq F(t) < 1$, lim F(t) = 1 as t-> inf and lim F(t) = 0 as t-> -inf

The pdf (probability) density function doesn't always exist!

P(A) = $\int_A p(t)dt$

$1 = P(\Omega) = \int_\Omega p(t)dt$

if p(t) exists, $p(t) = \frac{dF}{dt}$

Expectation (Expected Value)

$E(x) = \sum_{i=1}^{\infty} x_i p_i$

Ex. Bernoulli:

$E(X) = 0 \cdot (1 - p) + 1 \cdot p = p$

Cont Case (w pdf)

$E(x) = \int_\Omega t(t)dt$

Ex: Uniform density

p(t) = 1

0 1

$\int_0^1 p(t)dt = 1$

ω_t = pick t, 0<=t<=1

$X(\omega_t) = t$

$E(x) = \int_0^1 tp(t)dt = \int_0^1 tdt = \frac{1}{2}$

Variance

Given a random variable X, let $\mu = E(X)$. Then the variance of X var(x) = $E((x-\mu)^2)$

(σ is standard deviation)

$\sigma = \sqrt{var(x)}$

$E(x) = \int_0^1 (t-\frac{1}{2})^2 dt = \int_0^1 (t^2 - t + \frac{1}{4})dt = \frac{1}{12}$

$\sigma = \sqrt{\frac{1}{12}} = \frac{1}{2\sqrt{3}}$

Binomial Random Variables

Flip a coin n times. What is the probability of k heads?

$\Omega = \{\text{Collection of all sequences of n flips}\} = \{(F_1, F_2, \dots F_n)\}$

$\omega = (F_1, F_2, \dots F_n)$

Probability of heads = p

$Prob(\omega) = \prod_{i=1}^n p(J_i)$

So if $X(\omega)$ = number of heads

$(n\ c\ k)p^k(1-p)^{n-k} = P_{not} = (x=k) = n\ c\ k)p^k(1-p)^{n-k}$

$E(x) = \sum_{k=0}^n k(n\ c\ k)p^k(1-p)^k = \sum_{k=1}^n \frac{n!}{(k-1)!(n-k)!}p^k(1-p)^{n-k} =$

11/8/2021 Binomial Distribution

Binomial Distribution

Prob of k heads in n throws

$\binom{n}{k}p^k * (1-p)^{n-k}$

$X(\omega) = \#\text{ of heads}$

E(x) = np

Expectation is linear: $E(\alpha x + \beta y) = \alpha E(x) + \beta E(y)$

Let $x_j = \{ \begin{matrix} 0 & \text{if jth toss is tails} \\ 1 & \text{If jth toss is heads} \end{matrix} \}$

E(X_j) = O(1-p) + 1(p) = p

Since $X = \sum_{j=1}^n X_j$,

$E(X) = \sum_{j=1}^n E(X_j) = np$

Indicator random variable - In general, if A is an event, $I_A(\omega) = \{ \begin{matrix} 0 & \text{if } \omega \notin A \\ 1 & \text{if } \omega \in A \end{matrix} \}$

$E(I_A) = P(A)$

Variance:

$$E([x - E(x)]^2) = E(x^2 - 2xE(x) + E(x)^2) = E(x^2) - E(2xE(x)) + E(E(x)^2)$$
$$= E(x^2) - 2E(x)E(x) + E(x)^2 = E(x^2) - E(x)^2$$

Probability & Generating Functions

Given a discrete probability distribution,

$P(x = \alpha_j) = P_j \qquad (0 \leq P_j \leq 1, \sum_{j=0}^{\infty} P_j = 1)$

The corr gen Function is $F(x) = \sum_{j=0}^{\infty} P_j x^j$

Examples:

a) Bernoulli : $F_{Berr} = (1 - p) + px$

b) Binomial: $F_{Bin} = \sum_{j=0}^{n} \binom{n}{j} p^j (1-p)^{n-j} (x^j)$

Note: $F'(x) = \sum_{j=1}^{\infty} j P_j x^{j-1}$

$F'(1) = \sum_{j=1}^{\infty} j P_j = \sum_{j=0}^{\infty} j P_j = E(x)$

$$E(x^2) = \sum_{j=1}^{\infty} j^2 P_j = \sum_{j=1}^{\infty} (j)(j-1) P_j + j P_j$$

$$= F'(1) + \sum_{j=2}^{\infty} (j)(j-1) P_j = F'(1) + F''(1)$$

This shows that $var(x) = F''(1) + F'(1) - F'(1)^2$

Next Lab

Due Friday

Given Flip() which returns true w/ probability = 1/2.

Flip() and Flip() are independent (successive calls of flip are completely independent)

Write Flip(unsigned long a, unsigned long b) that returns true w/ probability equal to a/b

Expected number of calls to flip should be O(1)!

Hint: Ignore the constraint (the constraint being the O(1) restriction) when thinking about this.

11/9

Randomize Input - Can help uniformize response

Shuffle

```
#include<algorithm>
#include<random>
void shuffle(vector<int>&a){
    for(int i=1;i<a.size();++i){
        std::uniform_int_distribution<int> u(0,i);
        int j = u(gen);
        std::swap(a[i],a[j]);
    }
}
```

Think about

Generate every permutation (of n elements) n!

Can do nonrecursively or recursively

Generate all k element subsets of n element set $\binom{n}{k}$

Generate a random k-element subset of an n-element subset

O(k) time & space, sorted

Famous Random Algorithm - Prime Number Detection

RSA needs 500-1000 digit primes

Miller - Rabin Algorithm

Fermat's little Theorem

If p is prime, $1 \leq a < p$

$a^p \equiv a \mod p$

$a^{p-1} \equiv 1 \mod p$

But if $a^{p-1} \equiv 1$, p migt not be prime.

Try several as. Still not good enough.

Carmichael NUmbers: n not prime such that $a^{n-1} \equiv 1 \mod n \; \forall$ a rel prime to n

Ramanujan Story: 1729 is a number where you can express it as the sum of two cubes. He, however, did not realize that it is also a carmichael number

Algorithm

Test if p is prime: Write $p-1 = 2^k q$ where q is odd

Choose a, 1<a<p-1 randomly.

Ind mod p: $a^q, a^{2q}, a^{4q} \ldots a^{2^q}$

if $a^{2^k q} \not\equiv 1 \rightarrow$ p not prime

else for each $j \leq k$ s.t. $a^{2^j q} \equiv 1$

either j=0 --> p is probably prime

or j>0

if($a^{2^{j-1} q} = -1$) -> p is probably prime

if($a^{2^{j-1}q} \neq -1$) -> p is NOT prime

If procedure returns "Probably prime" then Prob(p is prime | a) > 0.25; Events are independent for different a

11/11

Generating a Random k-set

Choose k distinct elements from {1,2,...,n}

1. knnth: 1,2,3,...,n

1....j | j+1

a1 < a2....am

Accept or reject j+1?

k-m elements remain to be chosen from n-j possibilities

$\binom{n-j}{k-m}$ possible remaining subsets

O(n) time

O(k) space

a1<a2<...ak

How many of the $\binom{n-j}{k-m}$ subsets contain x= j+1

Accept w/ probability $\frac{\binom{n-j-1}{k-m-1}}{\binom{n-j}{k-m}} = \frac{k-m}{n-j}$

```
//algo to use if n/2 <= k <= n
VI knnth_rks(int n, int k){
    VI a(k);
    for(int m=0,j=1;m<k&&j<=n;++j){
        double xi = rv(); //0<=x<=1
        if((n-j)*xi<k-m){
            a[m++] = j+1;
        }
    }
    return a;
}
```

Floyd's Algorithm

O(k) time & space

Output not in orde

see picture

11/12/2021

By induction each of the $\binom{n-1}{k-1}$ subsets is equally likely

Choose x_k

What is the probability of getting $a_1 < a_2 < \ldots < a_k$

Case 1: $a_k \neq n$

Prob($x_k = a_1$) = $\frac{1}{n}$

...

Prob($x_k = a_k$) = $\frac{1}{n}$

P($a_1 < a_2 < \ldots < a_k$) =
$\sum_{i=1}^{k} P(a_1 < a_2 < \ldots < a_k | a_1 < \ldots a_{i-1} < a_{i+1} < \ldots < a_k) P(a_1 < \ldots a_{i-1} < a_{i+1})$
$= \frac{k}{n} \cdot \frac{1}{\binom{n-1}{k-1}} = \frac{1}{\binom{n}{k}}$

Case 2: $a_k = n$

P($a_1 < a_2 < \ldots < a_k$) =
$\sum_{i=1}^{k} P(a_1 < a_2 < \ldots < a_k | a_1 < \ldots a_{i-1} < a_{i+1} < \ldots < a_k) P(a_1 < \ldots a_{i-1} < a_{i+1})$
$= \frac{k}{n} \cdot \frac{1}{\binom{n-1}{k-1}} = \frac{1}{\binom{n}{k}} = \frac{k}{n} \frac{1}{\binom{n-1}{k-1}} = \frac{1}{\binom{n}{k}}$

This happened because $P(a_1 < a_2 < \ldots < a_k | a_1 < \ldots a_{i-1} < a_{i+1} < \ldots < a_k) = P(x_k = n)$
=P(choose at random) + P(random choice = prev x_i) = $\frac{1}{n} + \frac{k-1}{n} = \frac{k}{n}$

Another floyd's method (or more recentnly, Brent's method)

To check if a linked list cycles.

Floyd's method- have a fast pointer that moves up 2 each time and a slow pointer that moves up 1 each time. Keep going until pointers fall off the end of linked list (null ptr) and return that it doesn not cycle or keep going until pointers are equal to each other in which case return that linked list cycles

Pollard Rho

O($n^{\frac{1}{4}}$)

Hypothesize that the sequence $x_i \leftarrow ((x_{i-1})^2 - 1) \mod n$ where x_i is a random number between and 1 and n and it "behaves randomly"

Check if x_i, y_j have common factors - GCD is efficient | i is the slow pointer going from 1,2,3,4 and j is the slow pointer going 2,4,6,8,10, etc.

Use Floyd (or Brent) to determine if the sequence loops

Checks $gcd(x_i - x_{2^k}, n) \neq 1, n$

if not gcd(..) = diviser of proper

loop ~ $10 \cdot n^{\frac{1}{4}}$

Find a nontrivial divisor of n

x_o = random (2,n-2)

x_i = $(x_{i-1}^2 + 1) \mod n$ || $y_i = y_{i-1}^2 + 1 \mod p$

xx_i (fast pointer) = $(x_{i-1}^2 + 1) \mod n$

xx_i (fast pointer) = $(x_{i-1}^2 + 1) \mod n$

if d=kgcd$(x_i - xx_i, n)$<n

return d

if d==n

fail

11/15/2021

Birthday Paradox

Choosing m values randomly from {1,2,3,...n}, what is the probabily of a repeat > ?

Probability of m distinct values 1, $\frac{n-1}{n}$, $\frac{n-2}{n}$, ... $\frac{n-m-1}{n}$

Probability of repeat $1 - 1(1-\frac{1}{n})(1-\frac{2}{n})\dots(1-\frac{m-1}{n})$

x>0

$e^x = 1 + x + \frac{x^2}{2} + \frac{x^3}{6}\dots$

$< 1 + x + x^2 + x^3 + \dots$

= $\frac{1}{1-x}$

$$1 - x < e^{-x}$$

$1 - \frac{k}{n} < e^{-\frac{k}{n}}$

$(1-\frac{1}{n})(1-\frac{2}{n})\dots(1-\frac{m-1}{n}) < e^{-\frac{1}{n}}\dots e^{-\frac{m-1}{n}} = e^{-\frac{1}{n}\sum_{k=1}^{m-1}(k)} = e^{\frac{-m(m-1)}{2n}}$

$e^{\frac{-m(m-1)}{2n}} = 1 - \alpha$ (desired prob.)

$e^{\frac{-m(m-1)}{2n}} = 1 - \alpha$

$m(m-1) = 2nln(\frac{1}{1-a})$

$m \approx \sqrt{2ln(\frac{1}{1-a})}\sqrt{n}$

Nuber of iterations < $4\sqrt{p}$ < $4n^{\frac{1}{4}}$

Hash Tables

Goal: Abstract Data Type of Set

0. Create an empty set
1. add an element x tos et
2. query if set contains x
3. (optional) remove element x from set
4. We want search to be O(1)

Hash Function

Elements are positive integers

Can map a seq of bits to an int

$h(x) = b_{n-1}b_{n-2}\dots b_0 \to (\sum_{k=0}^{n-1}(b_k z^k)) \mod m$

Table has m buckets. Put x in bucket h(x)

What if $h(x_1) = h(x_2)$

Chaining vs Open addressing

Chaining	Open Addressing
Put x_1, x_2 into same bucket	

11/19/2021

Secondary Hashing

loc of ith probe = $(h_1(k) + ih_2(k)) \mod m)$

We want $h_1(k) + ih_2(k) \mod m = 0, 1, \ldots, m-1$ to reach every loc:

Simplest: Let $m = 2^l$, $h_2(k)$ is always odd

Recall Chaining:

Successful: $1 + \frac{\alpha}{2}$

Unsuccessful: $1 + \alpha$

Open Addressing: Expected # of probes

P(>0 probes) +P(>1 probe) + P(>2 probes) + ...

= p(1) + p(2 probes) + p(3 probes) + ...

+p(2 probes) + p(3 probes) + ...

+ p(3 probes) + ...

= 1P(1 probe) + 2P(2 probes) + 3 P(probes) +...

=$\sum_{k=1}^{\infty} kP(k \text{ probes})$

= E(# of probes)

What is P(>k probes)

= P(probing an occupied spot k times)

$= \frac{n}{m} \cdot \frac{n-1}{m-1} \cdot \frac{n-2}{m-2} \cdot \ldots \cdot \frac{n-k-1}{m-k+1}$

$< (\frac{n}{m})^k = \alpha^k$

(# of probles) < $\sum_{k=0}^{\infty} \alpha^k = \frac{1}{1-\alpha}$

Successful: E(# of probes) = $\frac{1}{n} \sum_{k=1}^{n} \#$ of probes to find kth key

$\approx \frac{1}{n} \sum_{k=1}^{n} E(\#$ of probes when k-1 keys in table$) = \frac{1}{n} \sum_{k=1}^{n} \frac{1}{1-\frac{k-1}{m}}$

=$\frac{1}{n} \sum_{k=1}^{n} \frac{m}{m-k+1} = \frac{m}{n} \sum_{k=1}^{n} \frac{1}{m-k+1}$

Let $H_m = 1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{m} \approx ln(m)$

$= \frac{m}{n}(\frac{1}{m} + \frac{1}{m-1} + \ldots + \frac{1}{m-n+1}) = \frac{m}{n}(H_m - H_{m-n}) = \frac{m}{n}(ln(m) - ln(m-n)) = \frac{m}{n} ln(\frac{m}{m-n}) = \frac{1}{\alpha} ln(\frac{1}{1-\alpha})$

Expected Hash Performance

	Chaining	Open Addressing
Unsuccessful	$1+\alpha$	$\frac{1}{1-\alpha}$
Successful	$1+\frac{\alpha}{2}$	$\frac{1}{\alpha}ln(\frac{1}{1-\alpha})$

Open addressing can't handle deletions!

11/22/21

Bicolor Towers of Hanoi

$h(n,t) = \#$ of moves to transfer n disks given t towers

h(n,t) = h(k,t) + h(n-k,t-1) where t is an upper portion of the initial tower

h(n,t) = min{(2h(k,t) + h(n-k,t-1)} with the restriction that $1 \leq k \leq n-1$

n	3	4
1	1	1
2	3	3
3	7	5
4	15	
5	31	

$2h(k,4)+h(3-k,3) = 2h(k,4)+2^{3-k}-1$

Assignment: implement this

H(n,from,to,aux)

h(nt,nd,aux) where nt is the number of towers, nd is the number of disks, and aux is the permutation

Other Recursion stuff

Recursion is not inherently inefficient

Non-Attacking Queens

```
int count =0;
int q[100];
void search(int r){
    if(r==n){
        ++count;
        return;
    }
    for(int c=0;c<n;c++){
        if(safe(r,c)){
            q[r]=c;//r is the row number
            search(r+1);
```

```
        }
    }
}
bool safe(int r, int c){
    //just a method to see if a certain row and column is safe from the other queens
}
```

13x13 --> 73712 solutions

Recursive tiem 0.53 sec

Nonrecursive time 0.76 sec

Exercise to show that recursive does not necessarily mean inefficient

Misc.

```
bool odd(int n);
bool even(int n){
    if(n==0){
        return true;
    }
    return odd(n-1);

}
bool odd(int n){
    if(n==0){
        return false;
    }
    return even(n-1);
}
```

11/29/2021

Dynamic Programming (DP)

```
//Exponential time algorithm
int fib(int n){
    if(n>2) return n;
    return fib(n-1)+fib(n-2);
}
```

Above is an exponential time algorithm for fibonacci, this is because you are repeating calcuations.

Ex calling fib(2) for both fib(4) and fib(3) in the recursion stack for fib(n) where n>2

Memoization

```
vi f;
int mem_fib(int n){
    if(f.size()>n){
        return f[n];
    }
    if(n<2){
        f.push_back(0);
        f.push_back(1);
        return n;
    }
    f.push_back(mem_fib(n-2)+mem_fib(n-1));
    return f(n);
}
```

On a side note, the only operators that gurantee evaluation of left arg before right are `? && || ,`

```
//Simple Dynamic Programming
f0=0;
f1=1;
for(int i=2;i<n;++i){
    f2=f0+f1;
    f0=f1;
    f1=f2;
}
return f2;
```

When a problem decomposes into smaller sub-problems having the same structure as the original, we save computation by storing previously computed sub-problems.

Top Down --> Recursive

Bottom Up --> DP

Example: m x n array of int. A path moves from top to bottom, choosing one of 3 cells below and adjacent to current cell, accumalating numner in cell. Maximize sum in one descent

Exponential number of paths -> $n * 2^{m-1}$ paths.

Sub-Problem: Start at row n

Store results in a table.

"Base" case: start at bottom row:

`sum[m-1][j]` = `a[m-1][j]`

`sum[i-1][j] = a[i-1][j]+max{sum[i][j-1],sum[i][j],sum[i][j+1]}`

Actual Application - Seam Carving

Crop a photo.

Score of path: see picture

11/30/2021

Knapsack

Choose a subset of items where each item has a value and a weight so that total weight > W and total value is maximized.

Edi Distance

Given 2 strings s0, s1 what is the min cost to turn s0 into s1 w/ a sequence of "edit operations," e.g. insert 1 char, change 1 char, delete 1 char, swap 2 adjacent characters, each having a certain cost?

Build a 2d table where c[i][j] is the cost of changing s0[0:i] to s1[0:j]

If |s0| = m, |s1|=n, then time = O(mn) and space = O(mn)

LCS: longest common subsequence

This was linux's diff does

12/2/2021

Egg Dropping

Egg breaks from floor all floors above f and never breaks under f

```
def getPartition(int eggs){
    return 1-x/(2x-1);
}
def binaryDrop(int flr,int ciel, int b){

}
def linearDrop(int flr,int b){

}
def initFunc(int b,int e){
    if(e==1){
        linearDrop(0,b);
    }
    else{
        binaryDrop(0,b,b);
    }
}
```

12/6/2021

** Make sure all labs are submitted by Wednesday 12/8**

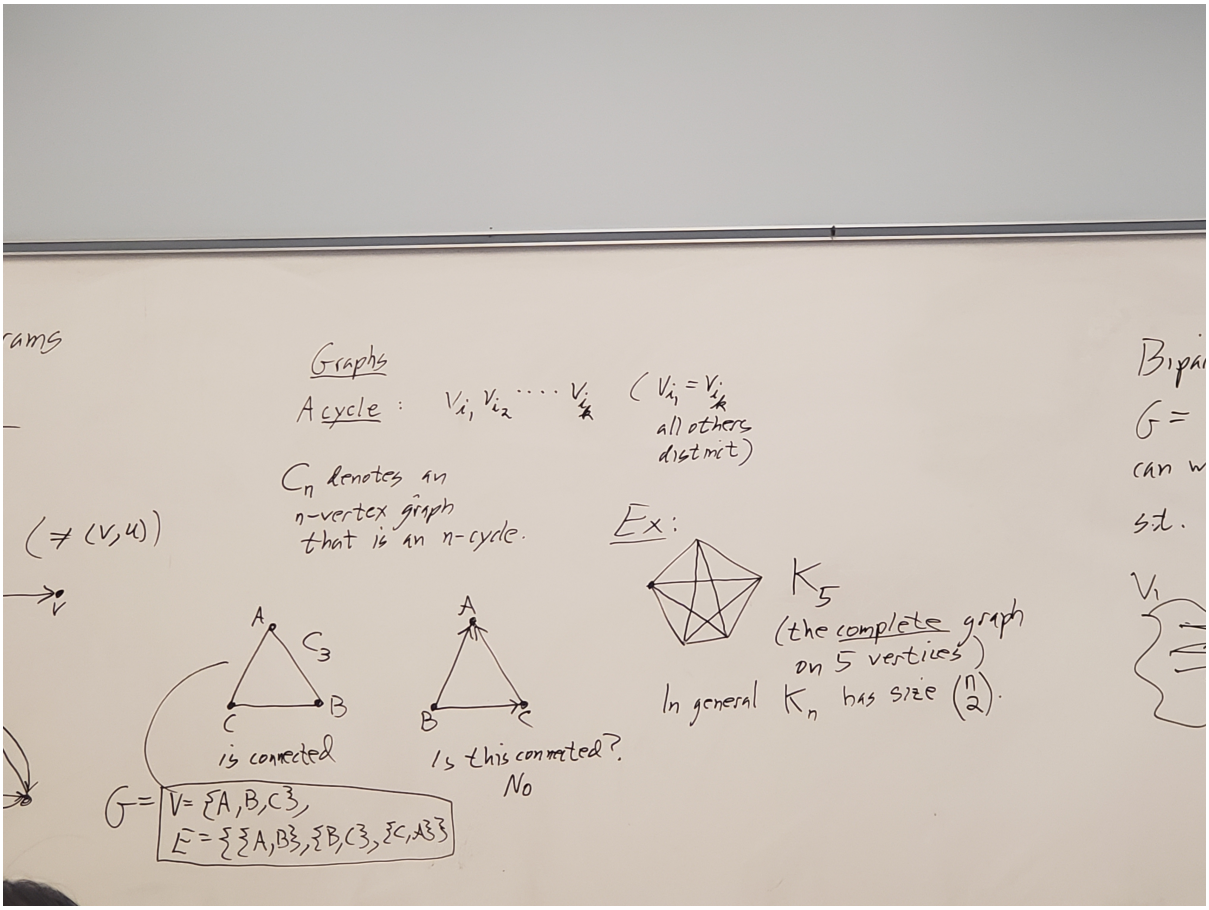
Graphs vs Digraphs

Graphs	vs	Digraphs
Edge = {u,v}		Arc = (u,v) ($\neq$ (v,u))
line segment between u and v		ray between u and v going from u to v
acts like a triangle with 3 vertices		acts like a triangle with a bunch of directionals

Graphs

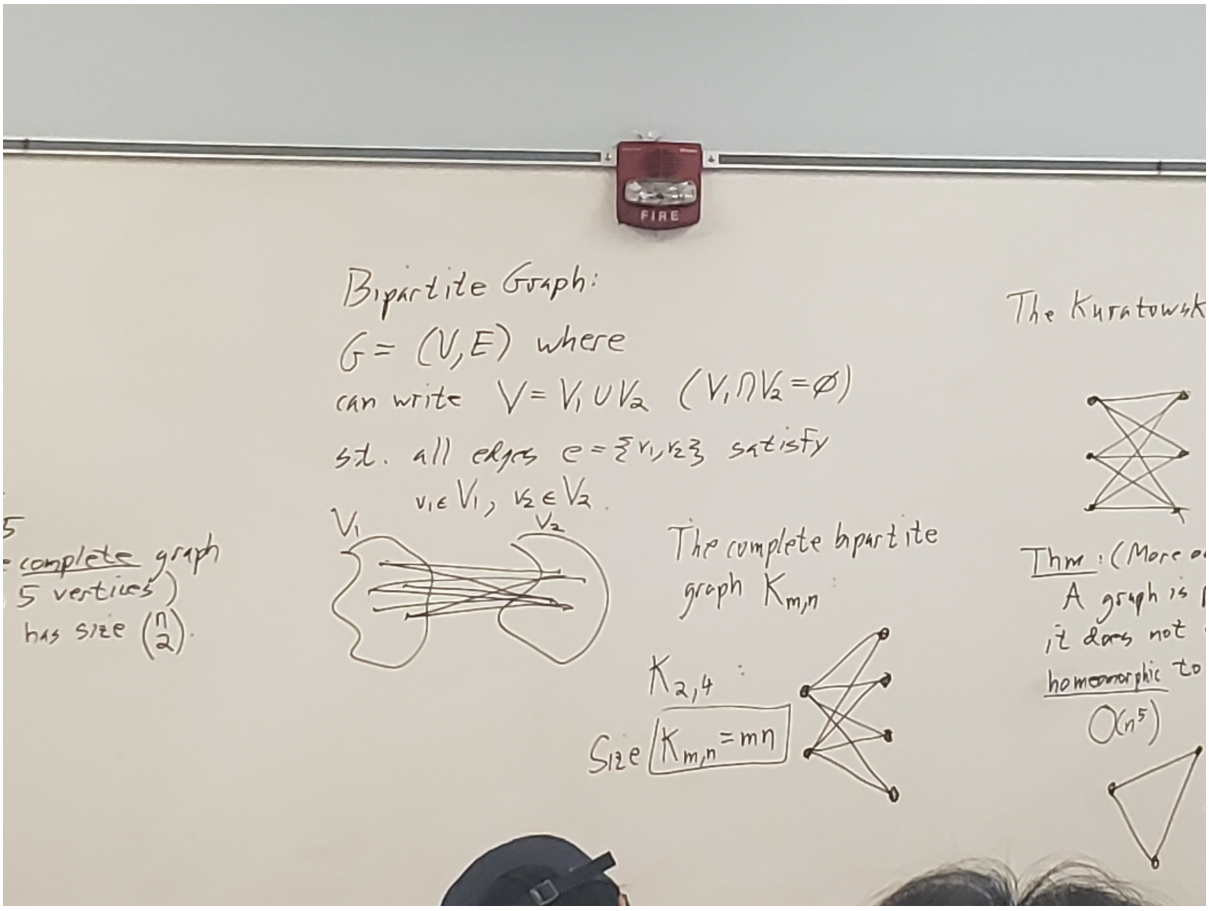
A cycle: $V_i, V_{i_2}, \ldots, V_{i_k}$ ($V_{i_1} = V_{i_k}$, all others distnict)

C_n denotes an n-vertex graph that is an n-cycle



Bipartite Graph:

G = (V,E) where you can write $V = V_1 \cup V_2$ ($V_1 \bigcap V_2 = \emptyset$)Such that all edges $e = \{v_1, v_2\}$ satisfy $v_1 \in V_1, v_2 \in V_2$

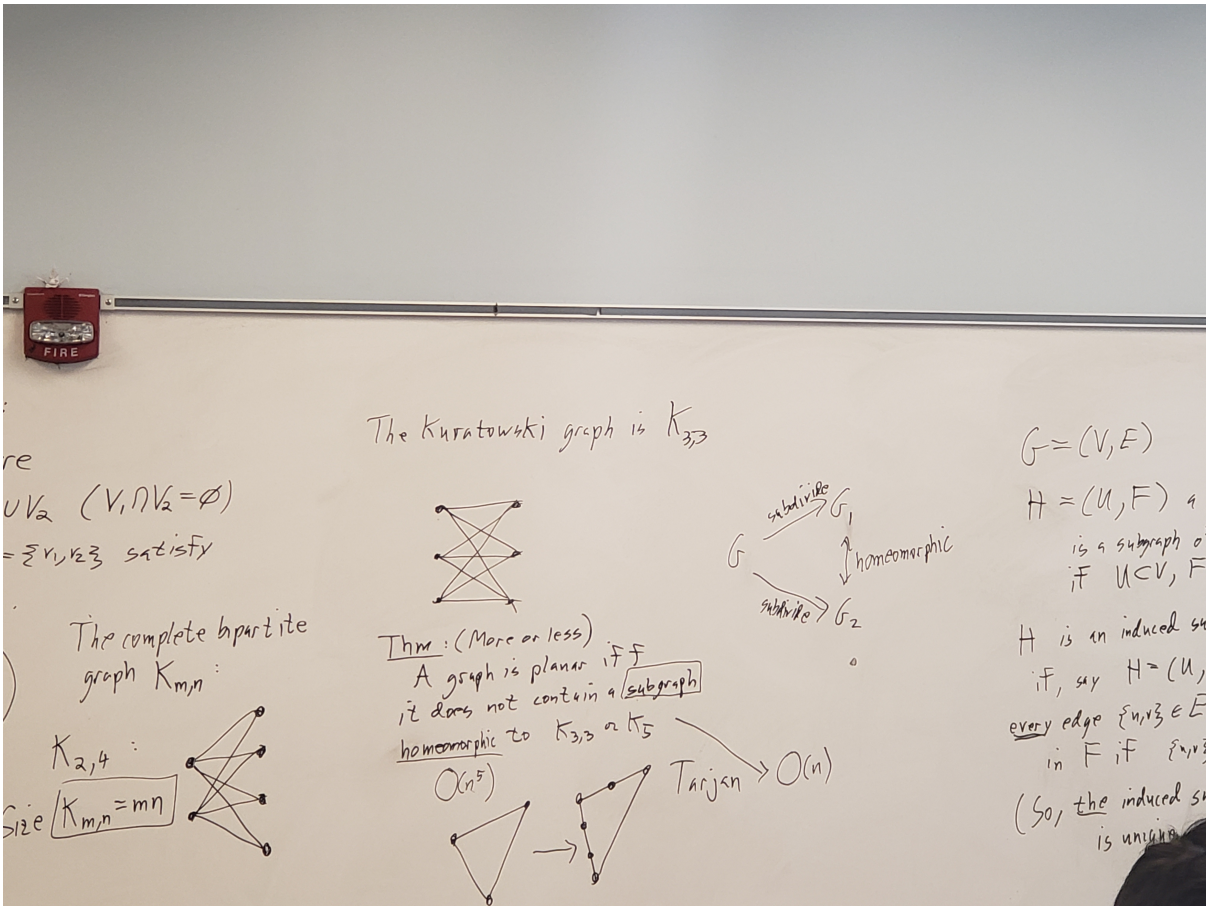


The Kuratowski Graph

Kuratowski graph is $K_{3,3}$

Theorem (More or less): A graph is planar iff it does not cointain a $\boxed{\text{subgraph}}$ homeomorphic to $K_{3,3}$ or K_5

$O(n^5)$

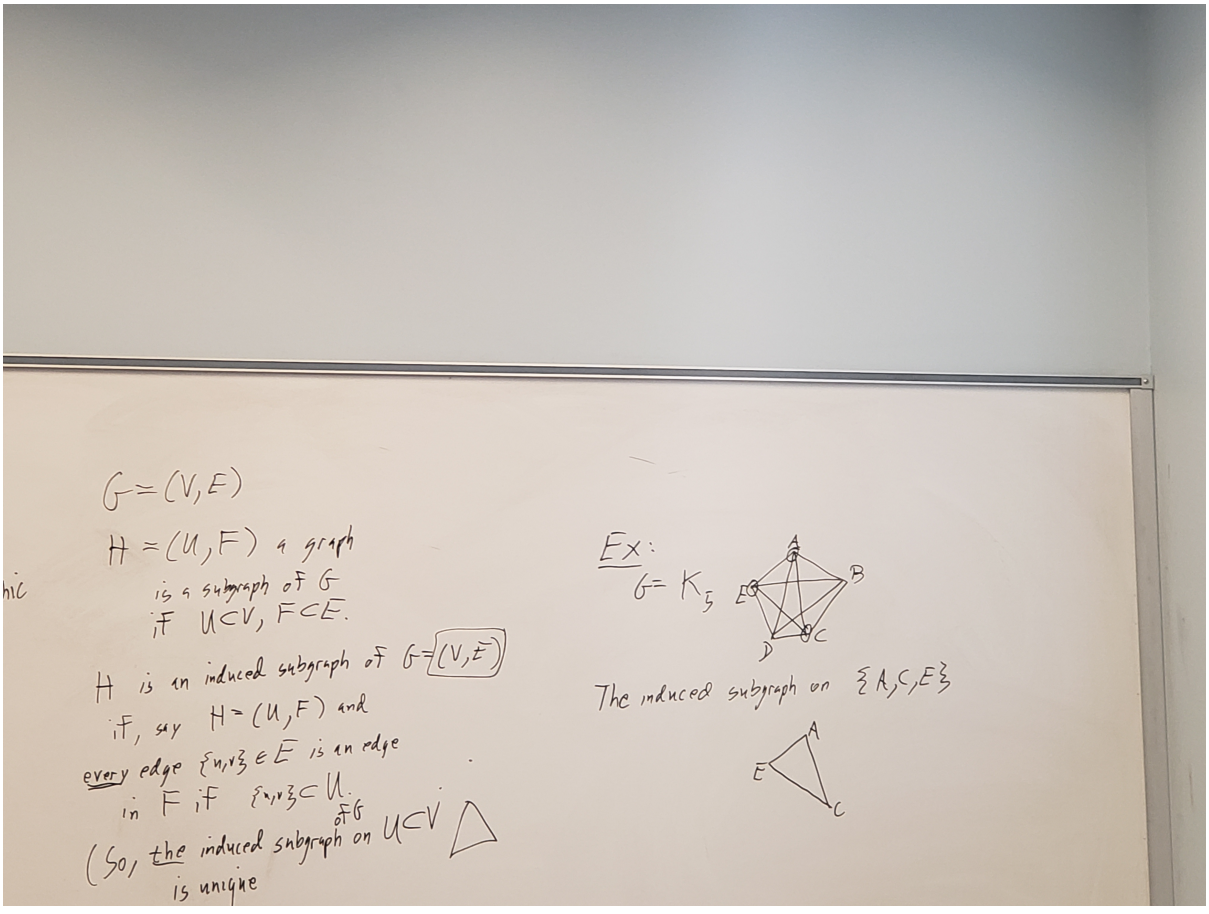


$G = (V, E)$

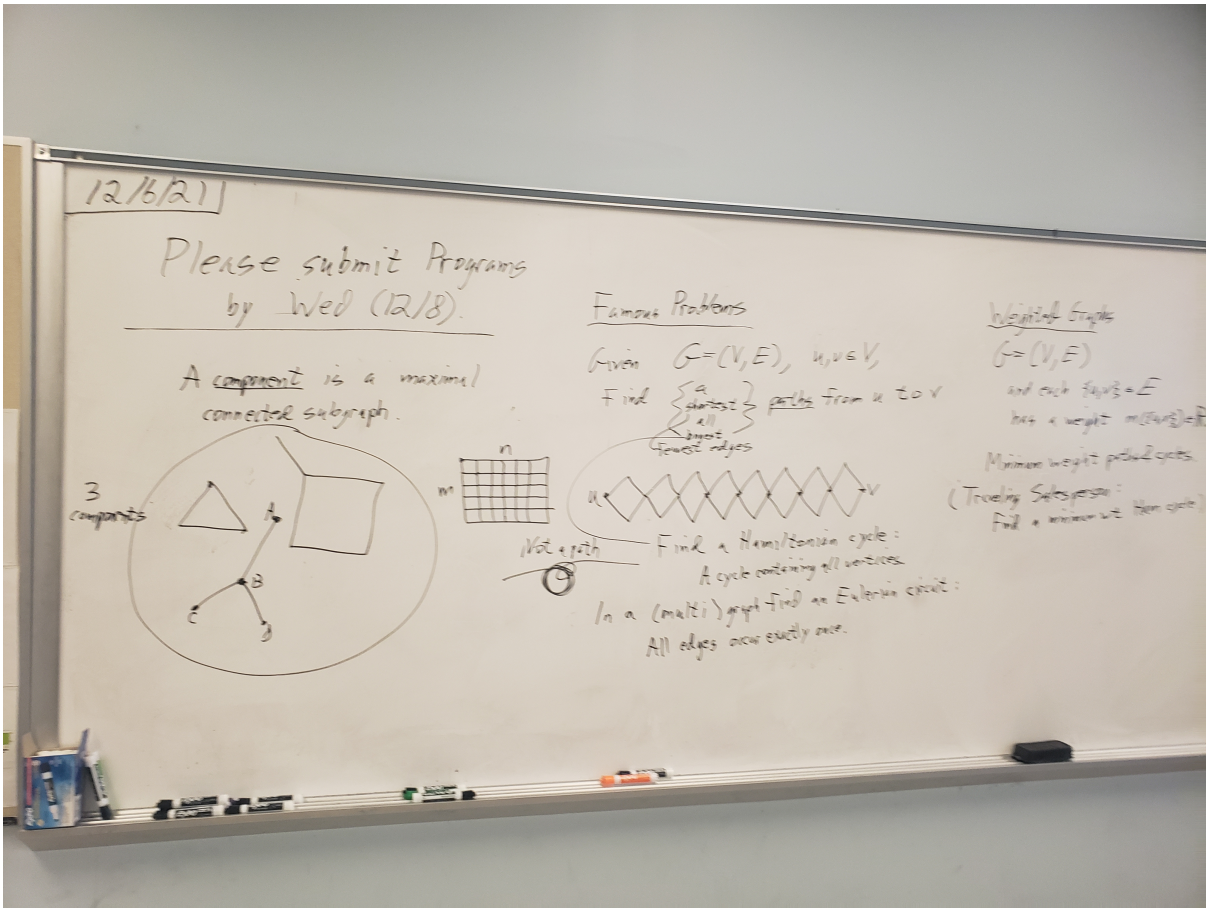
$H = (U, F)$ a graph is a subgraph of G if $U \subset V, F \subset E$

H is an induced subgraph of $G = (V, E)$ if, say $H = (U, F)$ and every edge $\{u, v\} \in E$ is an edge in F if $\{u, v\} \subset U$

So, the induced subgraph on $U \subset V$ is unique



A component is a maximal connected subgraph



Famous Problems

Given $G = (V, E),\ u, v \in V$,

Find a path from u to v, find the shortest path from u to v, find all paths from u to v, find the longest path from u to v.

A path can not repeat vertices/nodes

Shortest $\rightarrow$ Fewest edges

Find a Hamiltonian cycle: A cycle containing all vertices

In a (Multi) graph find an Eulerian circuit: All edges occur exactly once.

Weighted Graphs

$G = (V, E)$

and each $\{u, v\} \in E$ has a weight $m(\{u, v\}) \in \mathbb{R}$

Minimum Weight paths & cycles.

(Traveling Salesperson Person: Find a minimum weight Hamiltonian Cycle)