Spring 2019, **Final Exam Sol** (60 min In class Exam) **June 19 (Wed PM 12:00-1:00) at 32255**								**Sign**
Course	Discrete Math		**SKKU**	**Spring 2019**	Prof.	Sang-Gu LEE		
Class #	**GEDB007-44**	Major *(전공)*		Student No. *(학번)*			**Name**	

※ **Notice/Reference** http://matrix.skku.ac.kr/2019-album/

1. **Fill out the above boxes before you start this Exam.** *(학번, 이름 등을 기입하고 감독자 날인)*
2. **Honor Code:** *(시험 부정행위시 해당 교과목 성적이 "F" 처리됨은 물론 징계위원회에 회부될 수 있습니다.)*
3. **You have 60 minutes to complete this exam. Leave at least one seat between each other.**
4. **Please write clearly and properly. If we cannot read your writing, it will not be graded.**

Total Score (100 pt)	
Offline Exam 80	Participation 20

(20 + 20 + 20 + 20 + 20 = 100) A: 20% B: 30~40% C: 20~30% D, F: 20%

Part I. [2pt x 10= 20 Pts] True(T) or False(F). No explanations required.

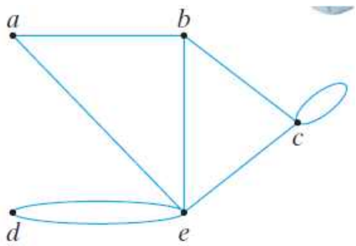
[F] 1. $\sum_{i=1}^{n} i \lg i = \Theta(n \lg n)$ **and** $1^3 + 2^3 + \cdots + n^3 = \Theta(n^4)$ **for all** $n \geq 1$. ($\sum_{i=1}^{n} i^2 \lg i = \Theta(n \lg n)$)

[T] 2. There are $8!/(3! \times 2!) = 3360$ **strings that can be formed by ordering the letters ILLINOIS.**

[T] 3. There are n **edges that are incident on a vertex in an** n**-cube.** (In n-cube, each vertex has n edges incident to it, since there are exactly n bit positions that can be toggled to get an edge.)

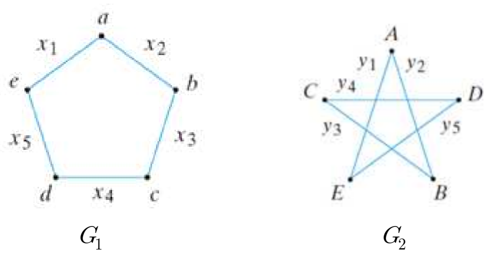
[F] 4. There are 8 eight-bit strings begin with 0 and end with 101. (8 eight-bit strings, 16 eight-bit strings)

[F] 5. The Incidence Matrix for this graph is (not Incidence, Adjacency Matrix)



$$\begin{array}{c|ccccc} & a & b & c & d & e \\ \hline a & 0 & 1 & 0 & 0 & 1 \\ b & 1 & 0 & 1 & 0 & 1 \\ c & 0 & 1 & 2 & 0 & 1 \\ d & 0 & 0 & 0 & 0 & 2 \\ e & 1 & 1 & 1 & 2 & 0 \end{array}$$

[F] 6. The graphs G_1 **and** G_2 **are not isomorphic.** (it is isomorphic)

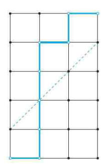


[T] 7. A graph is planar (Plane) if it can be drawn in the plane without its edges crossing.

[F] 8. The n**-cube is planar if** $n \leq 4$ **and not planar if** $n > 4$ (not 4, it should be 3)

[F] 9. A graph $G = (V, E)$ **is bipartite if there exist subsets** V_1 **and** V_2 **of** V**,** $V(G_1) \cap V(G_2) \neq \varnothing$**,**
$V(G) = V(G_1) \cup V(G_2)$ **each edge in** E **is incident on one vertex in** V_1 **and one vertex in** V_2**.**
(not $V(G_1) \cap V(G_2) \neq \varnothing$, it should be $V(G_1) \cap V(G_2) = \varnothing$)

[T] 10. There are C_n **non-isomorphic binary trees with** n **vertices where** $C_n = \frac{C(2n, n)}{n+1} = \frac{(2n)!}{(n+1)!\, n!} = \prod_{k=2}^{n} \frac{n+k}{k}$
is the n**-th Catalan number. (http://matrix.skku.ac.kr/sglee/catalan/catalan.htm)**



Part II. [20 Pts] State (Fill the boxes and/or state).

[5 Pts] 1. State more than *10 Discrete Mathematics Terminology, Concept, Definition, Theorem* that you have learned.

Inclusion-Exclusion Principle for n Sets, Fibonacci Number, Pascal's Triangle, Pigeonhole Principle,
Ackerman's Function, Linear Homogeneous Recurrence Relation, Binary and Hexadecimal Conversion,
Derangement, Dijkstra's Algorithm(A Shortest-Path Algorithm), Proof by Contradiction,
Mathematical Induction, Bacon Number, Matrix of the Relation. Binary Search Algorithm, Insertion Sort,
Euler Cycle, Hamiltonian Cycle(Gray Code Animation), Traveling Salesperson Problem (TSP),
Euler's Formula for Graph, Homeomorphic (G_1 and G_2 is isomorphic after some series reductions),
Constructing an Optimal Huffman Code, Minimal Spanning Tree, Depth-First Search(DFS) for a Spanning Tree,
Prim's Algorithm, Binary Tree, **Tree Traversals** (트리 운행-Preorder, Inorder, Postorder)

- A Hamiltonian cycle is a cycle which only visits each vertex once, except for the first and last vertex (which is the same).
- The Travelling Salesman problem is the problem of visiting every vertex in a graph once (and only once), with minimal costs (traversing edges).
- Dijkstra's algorithm is a dynamic programming algorithm for finding the minimum distance from one node in a graph, to all the other nodes in the graph.
- The adjacency matrix is the matrix made from the various edge-costs in a graph.
- Isomorphic graphs are graphs that are the same graph, just with the vertices moved around.
- Huffman code is a type of optimal prefix code that is commonly used for lossless data compression. We can compute Huffman code by using trees.
- Spanning trees span part of a graph, by connecting vertices.
- Breadth-First Search and Depth-First Search are tree traversal algorithms. BFS searches one level at a time, from left to right. DFS searches in a left-root-right-fashion.
- Binary search trees are trees where the value of a node's right child is larger than the parent node, while the left child's value is less than its parent's value. This makes for efficient searching of the tree.
- Prim's algorithm is a greedy algorithm to find the minimum spanning tree of a undirected and weighted graph, that does not update the cost to reach previously visited nodes. ...

[5 Pts] 2. State more than *5 tasks(prove, solve, or explain) that you can do* after you studied 9 Chapters of DM.

- I can State and use the addition, subtraction and multiplication principles on concrete problems
- Explain Catalan numbers
- Explain Fibonacci numbers
- Explain and use the Pigeonhole Principle on concrete problems
- Solve recurrence relation problems mathematically
- Solve linear homogeneous recurrence relations for constant coeffisients
- Explain and solve the Tower of Hanoi problem for any amount of discs
- Find Hamiltonian cycles
- Explain, classify and find simple cycles and paths
- Explain the Travelling Salesman problem
- Explain and use Dijkstra's algorithm to solve TSP-problems
- Find adjacency matrices
- Explain the principle of isomorphic and planar graphs
- Explain trees and tree traversal algorithms
- Use Breadth-First Search and Depth-First Search for tree traversal
- Explain what binary search trees are and state their time complexity for operations
- Use Prim's algorithm to find minimum spanning trees
- Explain decision trees and isomorphic binary trees ..

[5 Pts] 3. Explain the following algorithm as much as you can. http://matrix.skku.ac.kr/2018-DM/DM-Ch-5-Lab.html

This algorithm determines whether the integer $n > 1$ is prime.
If n is prime, the algorithm returns 0. If n is composite, the algorithm returns a divisor d satisfying $2 \le d \le \sqrt{n}$.
To test whether d divides n, the following algorithm checks whether the remainder when n is divided by d, $n \bmod d$ is zero.

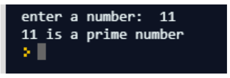
Input : n
Output : d
$is\_prime(n)\{$
    for $d = 2$ to $\lfloor \sqrt{n} \rfloor$
        if $(n \bmod d == 0)$
            return d
    return 0
    }

- Python code   https://sagecell.sagemath.org/   http://math3.skku.ac.kr/

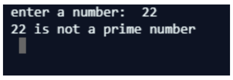
```
import math
n=int(input("enter a number: "))
for i in range(2,math.floor(math.sqrt(n))+1)://check if it has a divisor in the range[2,floor(sqrt(n))]
  if(n % i==0): //if there is no remainder, it means it has a divisor in this range
    print(str(n)+" is not a prime number")
    quit()
print(str(n)+" is a prime number") //if there is no divisor in the range, it is a prime
```

Practical image

Case A)



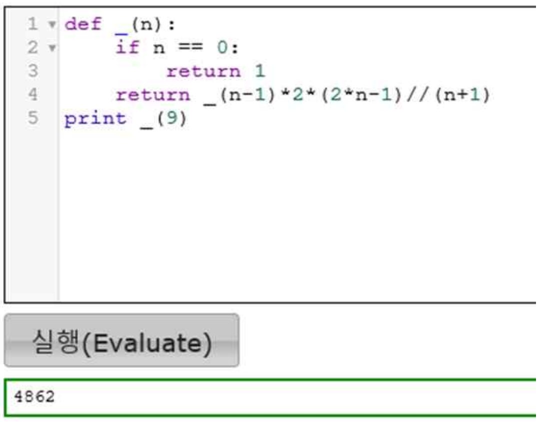
Case B)



[5 Pts] 4. Explain the following Python/Sage code for Catalan number as much as you can.

Python Code using recurrence relation formula:

```
import functools
@functools.lru_cache(maxsize=None)
def catalan(n):
    if n == 0:
        return 1
    return catalan(n-1)*2*(2*n-1)//(n+1)
n=int(input("enter a number"))
print(catalan(n))
```



Sage Code using recurrence relation formula:

```
def _(n):
    if n == 0:
      return 1
    return _(n-1)*2*(2*n-1)//(n+1)
print _(9)
```

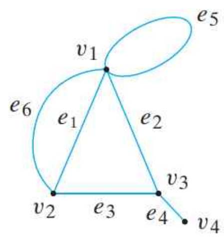
Part III. [4pt x 5= 20 Pts] Give your answer in the [Box]. Explanations are needed.

1. [Ch 6 Example 6.3.7, $C(k+t-1,\ t-1)=C(k+t-1,\ k)$] In how many ways can 12 identical mathematics books be distributed among the students Anna, Beth, Candy and Dan?

Solution

$$C(12+4-1,\ 4-1)=C(15,\ 3)=455.$$ **Answer = 455** ■

2. [Ch 8 P.407 Problem 11] For each graph $G=(V,E)$ in Exercises 11–13, find (1) V , E, (2) all parallel edges, (3) all loops, (4) all isolated vertices, and (5) tell whether G is a simple graph. Also, (6) tell on which vertices edge e_1 is incident.



Solution

(1) $V=\{v_1,\ v_2,\ v_3,\ v_4\}$, $E=\{e_1,\ e_2,\ e_3,\ e_4,\ e_5,\ e_6\}$
(2) In this graph, e_1 and e_6 are parallel edges.
(3) e_5 is a loop
(4) There is no isolated vertices.
(5) This graph is not a simple graph
(6) e_1 is incident on v_1 and v_2 .

3. Solve the recurrence relation (7.2.9) $a_n=5a_{n-1}-6a_{n-2}$ with initial conditions (7.2.10) $a_0=7$, $a_1=16$.

Solution

The characteristic equation of $a_n=5a_{n-1}-6a_{n-2}$ is $t^2=5t-6$ => $t^2-5t+6=(t-2)(t-3)=0 \Rightarrow t=2$ and $t=3$.

Let $S_n=2^n$ and $T_n=3^n$ $\Rightarrow$ $a_n=bS_n+dT_n=b2^n+d3^n$

$\Rightarrow 7=a_0=b2^0+d3^0=b+d$ and $16=a_1=b2^1+d3^1=2b+3d$
$\Rightarrow b=5$ and $d=2$.

Answer : $a_n=5\cdot 2^n+2\cdot 3^n$ **for** $n=0,1,\ldots$ ◀

4. [Ch 9 Trees, P.519 Exercises #4-#6]

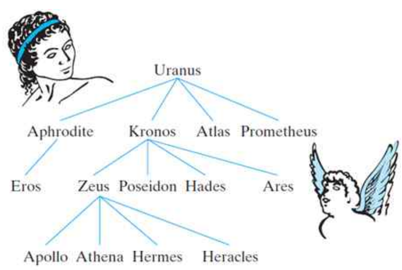


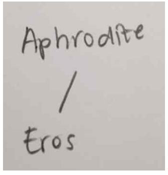
Figure 9.2.1 A portion of the family tree of ancient Greek gods.

(1) Find the parent of Poseidon. (2) Find the ancestors of Eros.
(3) Find the children of Uranus.

(4) Find the descendants of Zeus.
(5) Find the siblings of Ares.
(6) Draw the subtree rooted at Aphrodite.

Solution

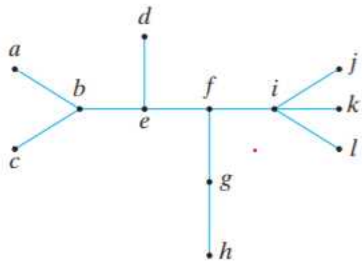
(1) Kronos
(2) Aphrodite, Uranus
(3) Aphrodite, Kronos, Atlas, Prometheus
(4) Apollo, Athena, Hermes, Heracles
(5) Zeus, Poseidon, Hades, Ares
(6)



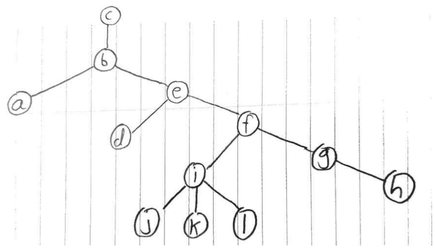
■

5. [Ch 9 Trees, Page 506, Exercise 1]

1. Draw the free tree as a rooted tree with root c.



Solution:



Part IV. [5pt x 4= 20 Pts] Find or give a sketch of your proof.

1. Find and explain the recurrence relations for Tower of Hanoi.

Tower of Hanoi

The Tower of Hanoi is a puzzle consisting of three pegs mounted on a board and n disks of various sizes with holes in their centers (see Figure 7.1.2). It is assumed that if a disk is on a peg, only a disk of smaller diameter can be placed on top of the first disk. Given all the disks stacked on one peg as in Figure 7.1.2, the problem is to transfer the disks to another peg by moving one disk at a time.

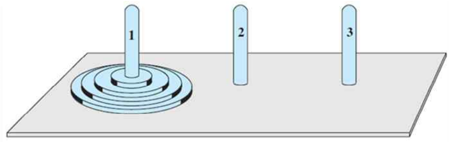


Figure 7.1.2 Tower of Hanoi.

Ans. $a_n = 2a_{n-1} + 1$.

Rule 1: you move one disk at a time. (Takes 1 sec).
" 2: You can put one disk on top of a larger one.
Let a_n be # moves required for n disks.
$a_1 = 1$.
For a_n, In order to move the largest disk (nth)
there must be nothing in C.
⇒ We need to move $n-1$ disks from A to B.
This takes a_{n-1} moves.
Move the largest from A to C (1 move)
Move the $n-1$ disks from B to C (a_{n-1} moves)
⇒ $a_n = 2 \cdot a_{n-1} + 1$.

$= 2^{n-1} + 2^{n-2} + 2^{n-3} + \cdots + 2 + 1$

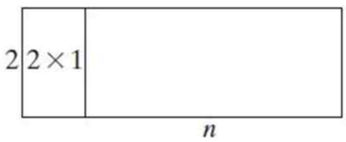
$= 2^n - 1$

Geometric sum:
$a + ar^1 + ar^2 + \ldots + ar^n$
$= a(r^{n+1}-1) / (r-1)$

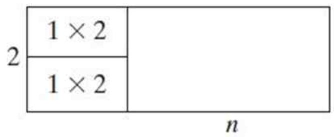
2. Give your combinatorial argument on

> **Suppose that we have a $2 \times n$ rectangular board divided into $2n$ squares. Let a_n denote the number of ways to exactly cover this board by 1×2 dominoes. Show that the sequence $\{a_n\}$ satisfies the recurrence relation**
>
> $$a_n = a_{n-1} + a_{n-2}.$$
>
> **Show that $a_n = f_{n+1}$, where $\{f_n\}$ is the Fibonacci sequence.**

Solution If the first domino is placed as shown in (a), there are a_{n-1} ways to cover the board that remains.



(a)



(b)

If the first two dominoes are placed as shown in (b), there are a_{n-2} ways to cover the board that remains.

It follows that $a_n = a_{n-1} + a_{n-2}$ by inspection.

Since a_n satisfies the same recurrence relation as the Fibonacci sequence and $a_0 = 0$ and $a_1 = 1$ and $a_2 = 2$,
$a_n = f_{n+1}$ where $\{f_n\}$ is the Fibonacci sequence. ■

3. Give your combinatorial argument on Pascal's Identity $C(n,\ k)=C(n-1,\ k)+C(n-1,\ k-1)$ **for** $1 \le k \le n$.

Pf)

Thm (Pascal's identity)

$$\binom{n}{k}=\binom{n-1}{k}+\binom{n-1}{k-1}$$

Pf) $\binom{n}{k}$ = the number of k-subsets of $\{1, 2, \ldots, n\}$

= (the number of k-subsets containing n) + (the number of k-subsets not containing n)

= (the number of $(k-1)$ subsets of $\{1, 2, \ldots, n-1\}$) + (the number of k-subsets of $\{1, 2, \ldots, n-1\}$)

$=\binom{n-1}{k}+\binom{n-1}{k-1}$ ◂

이 성질을 조금 쉽게 이해 해 보겠습니다. ${}_nC_r$ 이라는 것은 n개의 공에서 r개를 고르는 경우의 수입니다. 그러면 저는 n개의 공에서 1개를 마음속으로 골랐다고 생각 해 보겠습니다. 여기서 n개의 공은 모두 같은 공입니다. 그러면 최종 선택한 r개의 공 중 제가 마음속으로 고른 공이 들어있는 경우의 수가 있고 제가 고른 공이 있지 않은 경우의 수도 있을 것입니다. 제가 마음속으로 고른 공이 포함 되지 않은 경우는 ${}_{n-1}C_r$일 것입니다. 제가 마음속으로 고른 공이 포함 된 경우는 그 공이 이미 포함 되었다는 가정 하에 ${}_{n-1}C_{r-1}$ 이 될 것입니다. 따라서 ${}_{n-1}C_r+{}_{n-1}C_{r-1}={}_nC_r$ 이 됩니다. ■

4. [Catalan Number] How many routes are there from the lower-left corner of an $n\times n$ **square grid to the upper-right corner** if we are restricted to traveling only to the right or upward and if we are allowed to touch but not go above a diagonal line from the lower-left corner to the upper-right corner**?**

Solution

Let G_n **denote the number of good routes.** B_n **denote the number of bad routes.** $G_n+B_n=C(2n,\ n)$

The number of bad routes is equal to the number of $(n+1)\times(n-1)$ **routes.**

Thus the number of good routes is

$$G_n = C(2n,\ n)-B_n=C(2n,\ n)-C(2n,\ n-1)=\frac{(2n)!}{n!\,n!}-\frac{(2n)!}{(n-1)!\,(n+1)!}$$

$$=\frac{(2n)!}{n!\,(n-1)!}\left(\frac{1}{n}-\frac{1}{n+1}\right)=\frac{(2n)!}{n!\,(n-1)!}\cdot\frac{1}{n(n+1)}=\frac{(2n)!}{(n+1)n!\,n!}=\frac{C(2n,\ n)}{n+1}$$ ◂

Part V. [20 Pts] Spring 2019, DM Final Exam (Participation) (20 point / 100)

Student No.		Name	

<Fill this form, Print it, Bring it and submit it just before your Exam.>
(시험 전에 프린트하여, 빈칸을 채워서 제출하거나, 시험 중에 확인하여 채워서 시험 시간 중에 제출하면 됩니다.)

A. (10 pt) PBL(Problem-Project Based Learning) Participations. (Quantity !!!)

QnA Participations Numbers <Check yourself> : each weekly (From Friday - next Thursday)

- Week 1: Week 2: Week 3: Week 4:
- Week 5: Week 6: Week 7: Week 8:
- Week 9: Week 10: Week 11: Week 12:
- Week 13: Week 14: Week 15: (Total = entries)

Total number (Q: , A:) (80+ = A, ... , 39- = D)

■ Number of online attendances: ( ) / (80+) (1-15 week)
Off-line attendance: ( ) / (27+) (1-15 week)
Absences:

1. What is your most important contribution (or finding) that you shared with others in QnA. (Quality !!!)

2. Write what you especially talked/learned from our Action Learning/PBL class.

B. Write names of YOUR Team members and Team Leader.

■ Team Number and Leader :
■ Team Members :

3. Write the title of your project and your role in your Team and the most interesting things that you learned from your project and presentation.

4. (1pt, Bonus) Write anything you like to tell me. (What are things that you have learned and recall well from QnA and HW/PBL participation?, [개인/동료와 같이 LA 강좌를 (PBl/Filpped/Action learning) 학습 하면서 배우거나 느낀 점은?]

* Dear my DM students, Hope you can have a very nice Summer Break! ^^

[Scratch page]