Combinatorics and Applications

Ephram Chun

Mentor: (Victor)Jung Soo Chu

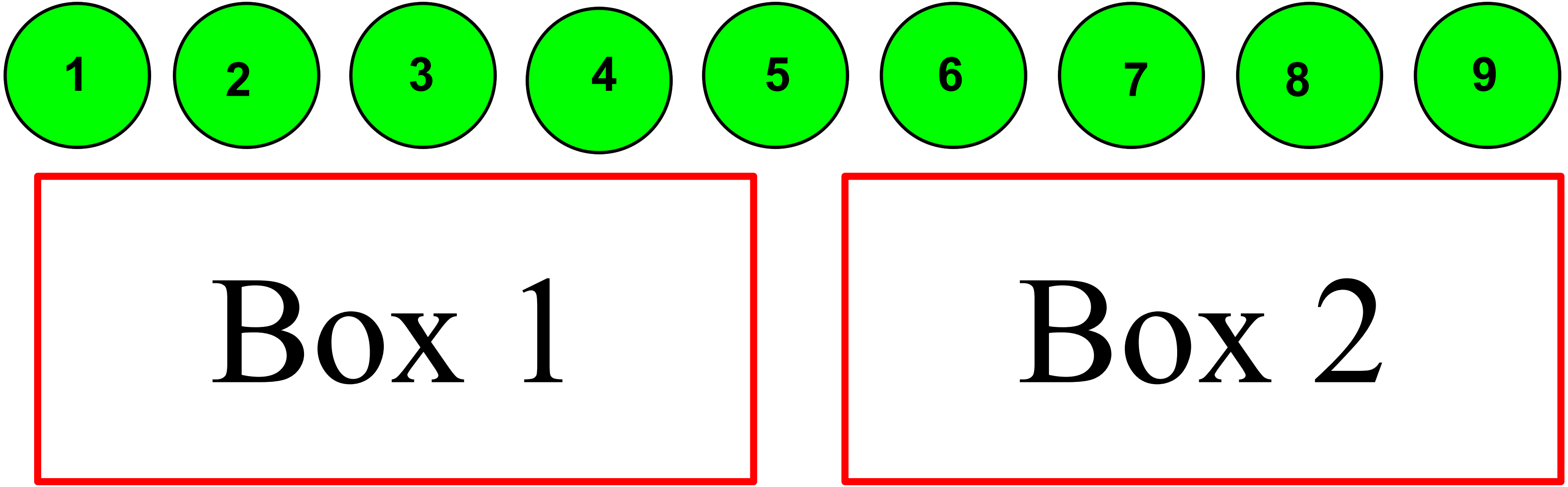
Overview

Bona, Miklos. *A Walk Through Combinatorics Fourth Edition*

- ❖ PigeonHole Principle
- ❖ Induction
- ❖ Basic Counting Problems
- ❖ Graph Theory
- ❖ Combinatorial Algorithms and Game Theory
- ❖ Applications of Combinatorics in Classroom Assignment Optimization

Basic PigeonHole Principle

Pigeonhole Principle: *Let n and k be positive integers, and let n > k. Suppose we have to place n identical balls into k identical boxes. Then there will be at least one box in which we place at least two balls.*



Box 1

Box 2

Proof of PigeonHole Principle

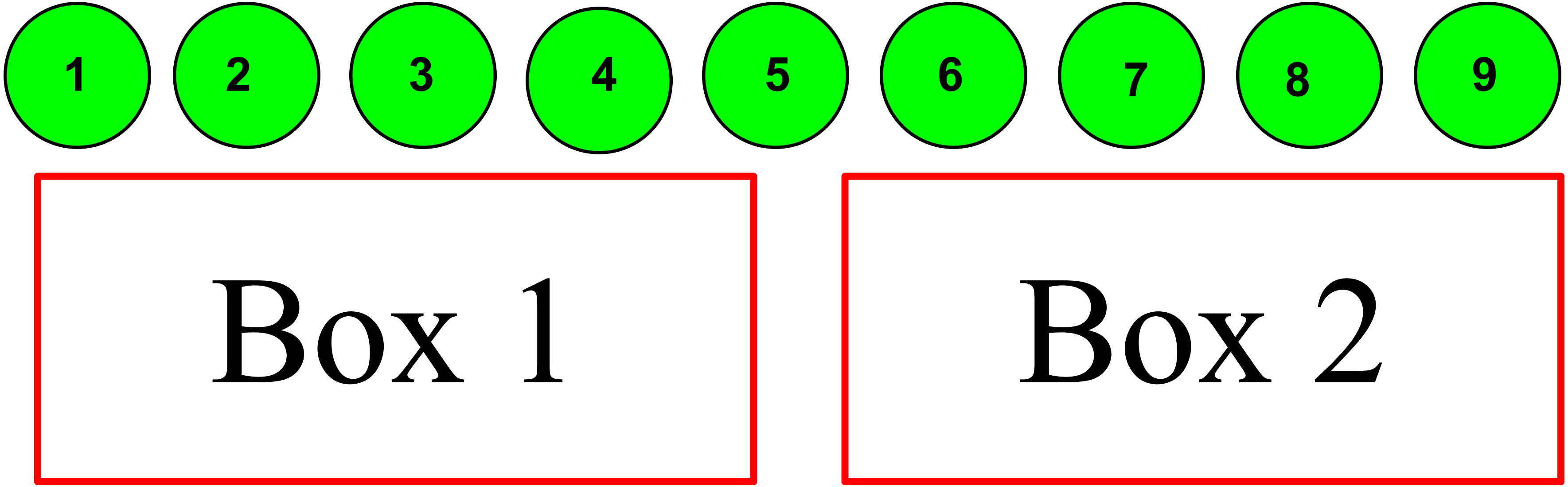
Proof:

We proceed with proof by contradiction. Let us assume that there is no box with at least two balls. Then each of the k boxes must have either 0 or 1 balls in it. Then the maximum number of balls that can be in the k boxes is $k*1=k$.

However, we are given that $n > k$ thus we have reached a contradiction thus our assumption that there is no box with at least two balls must be false. QED

Generalized PigeonHole Principle

Generalized PigeonHole Principle: Let n, m, and r be positive integers so that n>r*m, and let us distribute n identical balls into m identical boxes. Then there will be at least one box into which we place at least r +1 balls.



Box 1

Box 2

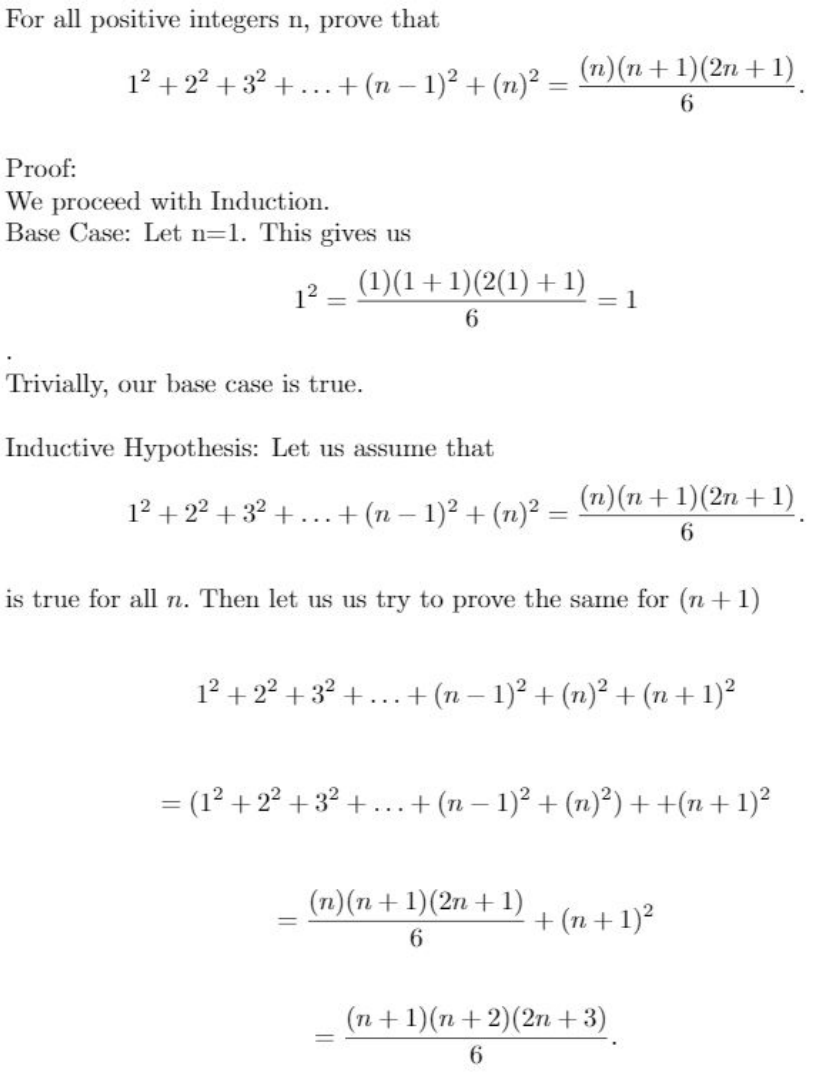
Generalized PigeonHole Principle Proof

Proof: We proceed with proof by contradiction. Let us assume that there is no box where we place at least $r +1$ balls in. That means we can place an integer between 0 and r balls in each of the m boxes. Thus, the maximum possible number of balls we can have is $r*m$.

However, it was given to us that $n>r*m$ thus a contradiction thus there must be at least one box in which we place at least $r +1$ balls. QED.

Weak Induction

1. **Initial Step:** Prove that the statement is true for the smallest of n for which it is defined.
2. **Induction Step:** Prove that from the fact that the statement is true for n, it follows that the statement is also true for n+1.



For all positive integers n, prove that

$$1^2 + 2^2 + 3^2 + \ldots + (n-1)^2 + (n)^2 = \frac{(n)(n+1)(2n+1)}{6}.$$

Proof:
We proceed with Induction.
Base Case: Let n=1. This gives us

$$1^2 = \frac{(1)(1+1)(2(1)+1)}{6} = 1$$

.

Trivially, our base case is true.

Inductive Hypothesis: Let us assume that

$$1^2 + 2^2 + 3^2 + \ldots + (n-1)^2 + (n)^2 = \frac{(n)(n+1)(2n+1)}{6}.$$

is true for all n. Then let us us try to prove the same for $(n+1)$

$$1^2 + 2^2 + 3^2 + \ldots + (n-1)^2 + (n)^2 + (n+1)^2$$

$$= (1^2 + 2^2 + 3^2 + \ldots + (n-1)^2 + (n)^2) + +(n+1)^2$$

$$= \frac{(n)(n+1)(2n+1)}{6} + (n+1)^2$$

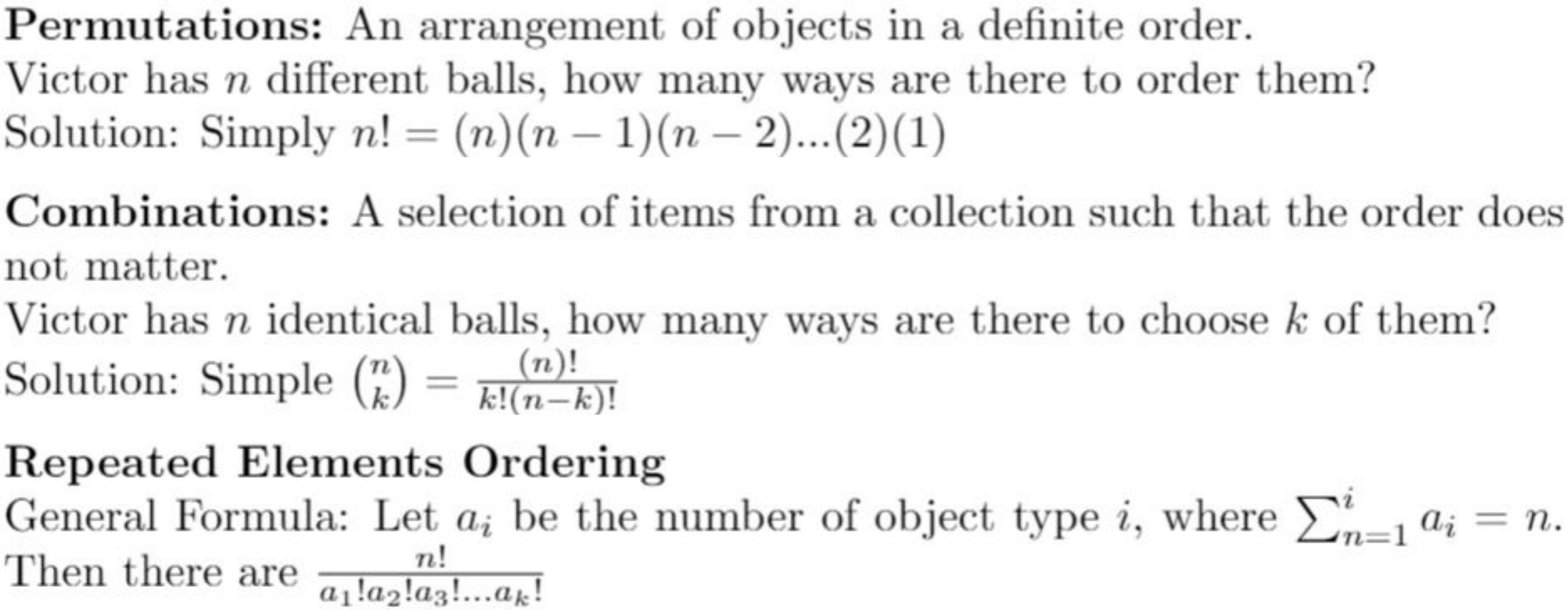
$$= \frac{(n+1)(n+2)(2n+3)}{6}.$$

Strong Induction

Strong Induction is split into 2 steps.

1. **Initial Step:** Equivalent to Weak Induction we prove that the statement is true for the smallest value of n for which it is defined.
2. **Induction Step:** Prove from the fact that the statement is true for all integers less than n+1, it follows that the statement is also true for n+1.

Basic Counting Problems



Permutations: An arrangement of objects in a definite order.
Victor has n different balls, how many ways are there to order them?
Solution: Simply $n! = (n)(n-1)(n-2)...(2)(1)$

Combinations: A selection of items from a collection such that the order does not matter.
Victor has n identical balls, how many ways are there to choose k of them?
Solution: Simple $\binom{n}{k} = \frac{(n)!}{k!(n-k)!}$

Repeated Elements Ordering
General Formula: Let a_i be the number of object type i, where $\sum_{n=1}^{i} a_i = n$.
Then there are $\frac{n!}{a_1!a_2!a_3!...a_k!}$

Graph Theory

Definition 1.1: A **graph** is a diagram from a set of points and lines that connect some pairs of those points.

Definition 1.2: The **vertices** also known as nodes are endpoints of an edge

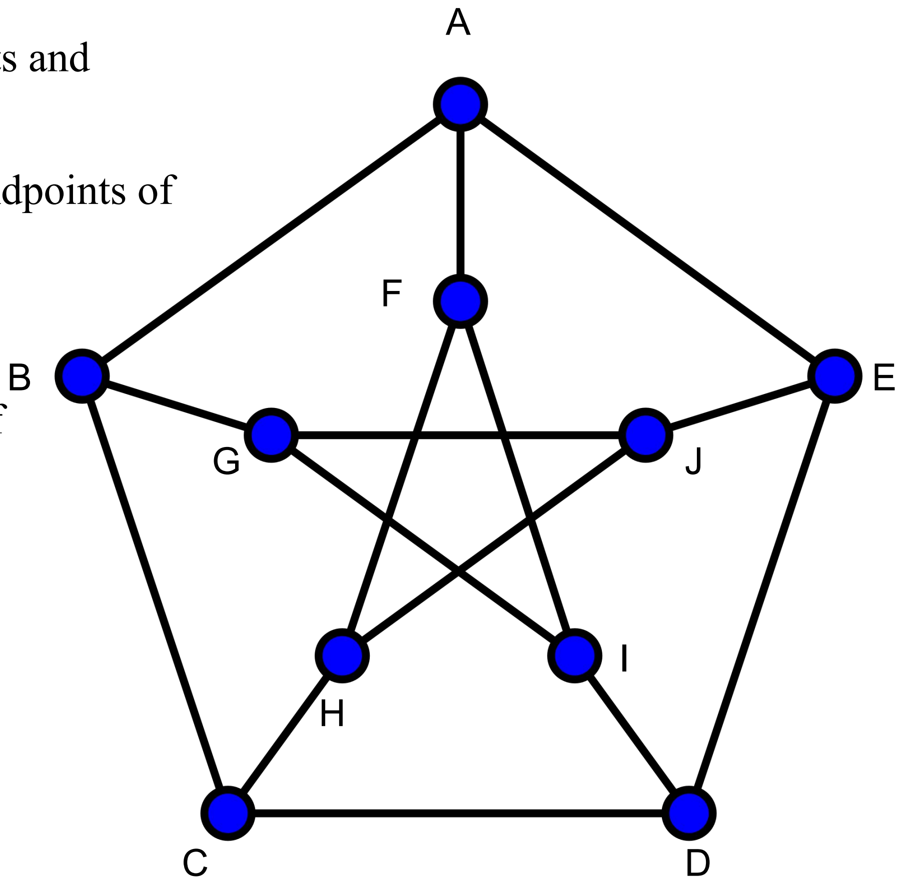
Definition 1.3: The **edges** are the lines that connect the vertices to each other

Definition 1.4: The **degree of a vertex is** the number of edges connected to a vertex

Definition 1.5: A **walk** is where no edge is repeated but a vertex can be repeated

Definition 1.6: A **cycle** is a closed trail that does not touch any vertex twice except the starting and ending vertice.

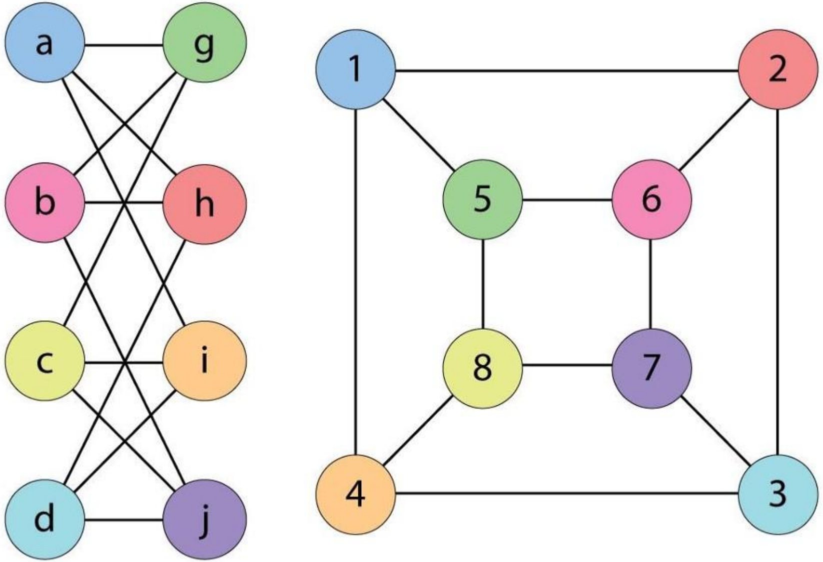
Definition 1.7: A **connected graph is** a graph where there is a path between any 2 vertices



Isomorphisms

Bijection: A correspondence between 2 sets where each element of one set corresponds with an element of the other set.

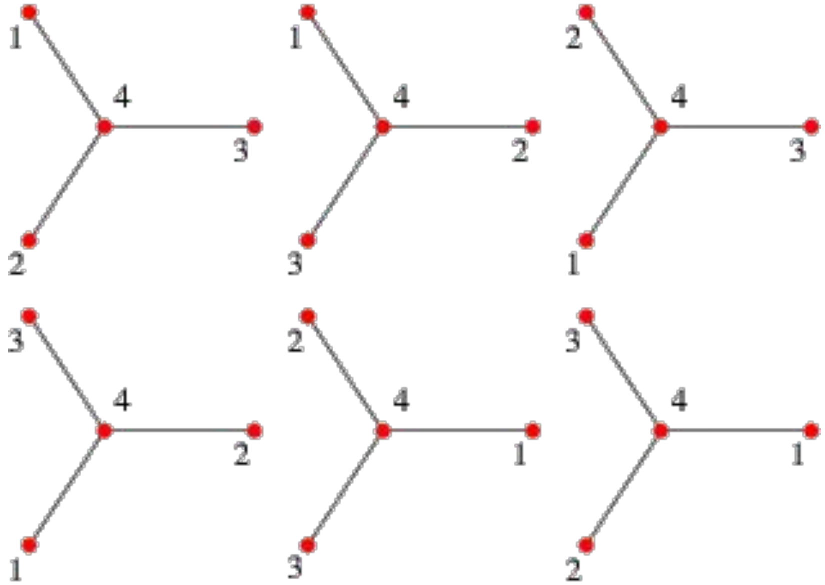
Isomorphism: Graphs G and H are isomorphic if there is a bijection f from the v(G) to v(H) so that the number of edges between any pair of vertices X and Y of G is equal to the number of edges between vertices f(X) and f(Y) of H.



a → 1
b → 6
c → 8
d → 3
g → 5
h → 2
i → 4
j → 7

Automorphism

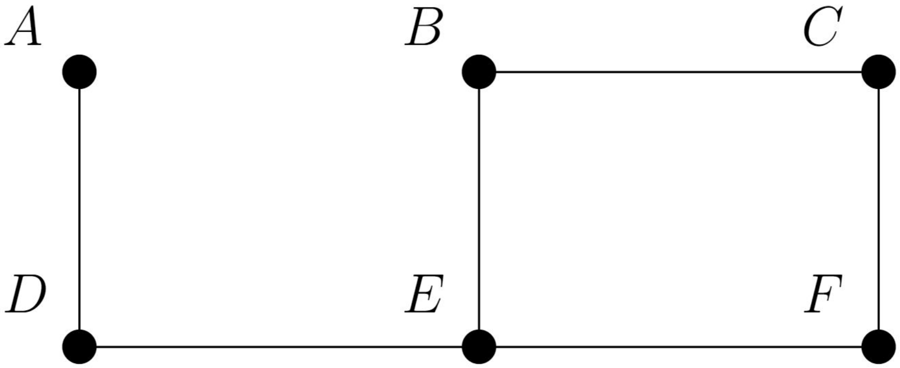
Automorphism: An automorphism of graph G is an isomorphism between G and G itself. That is, the permutation f of the vertex set of G is an automorphism of G if for any two vertices x and y of G, the number of edges between x and y is equal to the number of edges between f(x) and f(y). We are mapping the object onto itself while preserving the structure.

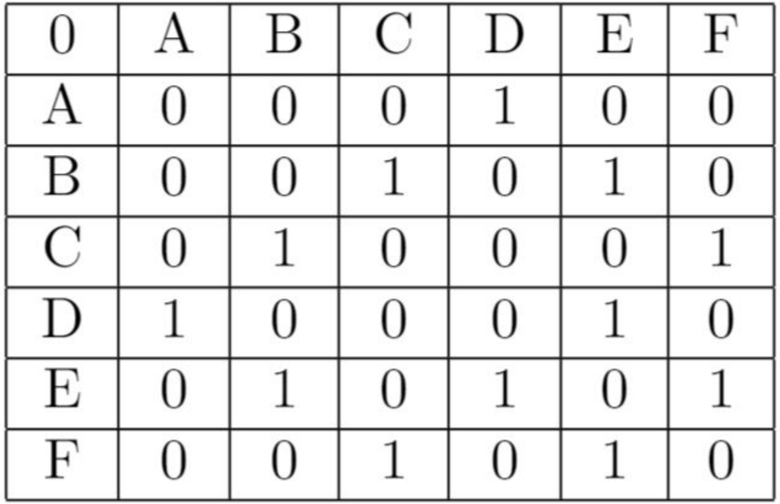


Adjacency Matrix

Adjacency Matrix: Let G be an undirected graph on n unlabeled vertices, and define an $n \times n$ matrix $A = A_G$ by setting $A_{i,j}$ equal to the number of edges between vertices i and j.

$A_{i,j}$ is the number of edges from i to j.



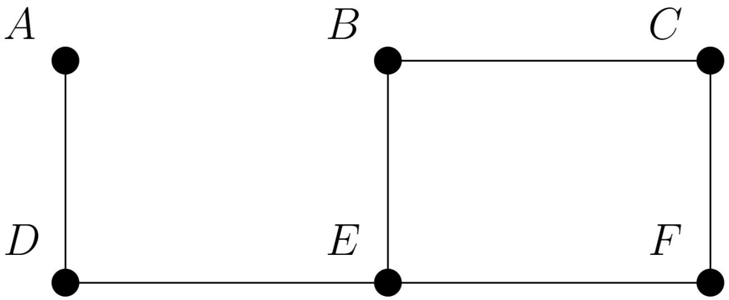


0	A	B	C	D	E	F
A	0	0	0	1	0	0
B	0	0	1	0	1	0
C	0	1	0	0	0	1
D	1	0	0	0	1	0
E	0	1	0	1	0	1
F	0	0	1	0	1	0

Matrix Multiplication

Theorem 10.16 Let there be a graph with n labeled vertices and an adjacency matrix A. Then $A^k_{i,j}$ is the number of paths from i to j that are length k.

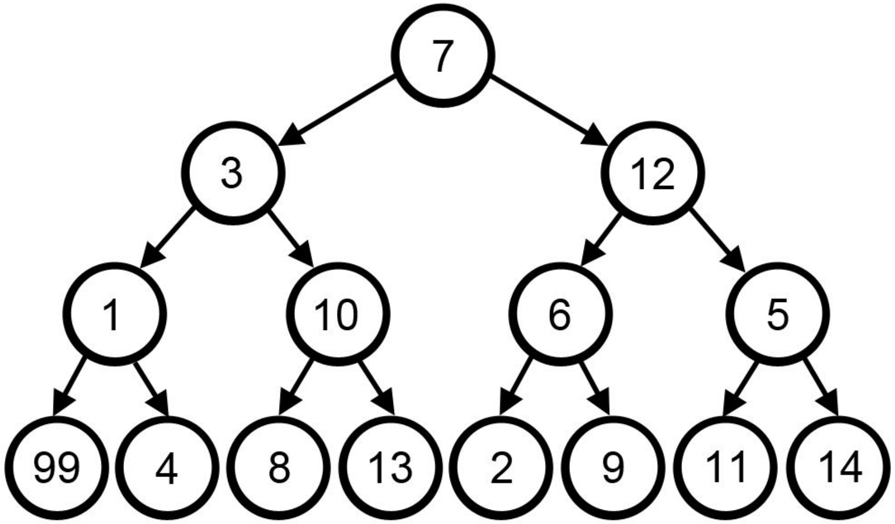
0	A	B	C	D	E	F
A	0	0	0	1	0	0
B	0	0	1	0	1	0
C	0	1	0	0	0	1
D	1	0	0	0	1	0
E	0	1	0	1	0	1
F	0	0	1	0	1	0



$$\begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 2 & 0 & 1 & 0 & 2 \\ 0 & 0 & 2 & 0 & 2 & 0 \\ 0 & 1 & 0 & 2 & 0 & 1 \\ 1 & 0 & 2 & 0 & 3 & 0 \\ 0 & 2 & 0 & 1 & 0 & 2 \end{pmatrix}$$

Greedy Algorithm

The **Greedy Algorithm** is an algorithm that will make decisions based on the immediate reward and never returns to a previously made decision.



In this example, the greedy algorithm is trying to choose numbers that will add up to a maximum value and will make choices at each step:

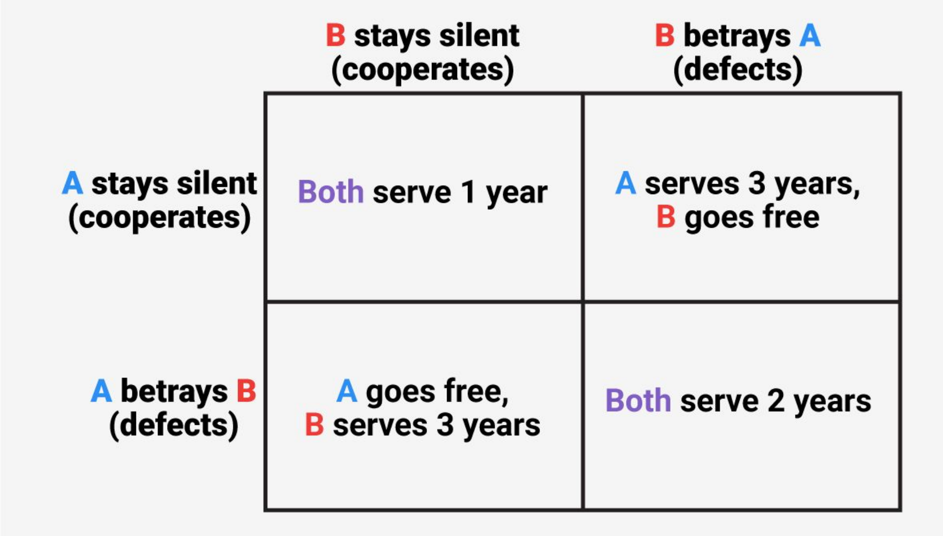
12 > 3 so it chooses 12
6 > 5 so it chooses 6
9 > 2 so it chooses 9

Dynamic Programming

- **Dynamic Programming** is an optimization over recursion, where it stores the results of subproblems and an optimization is found by a combination of these results.
- A **subproblem** is where we look at the smallest part of a problem and solve it there and then we use that to solve the next subproblem which is the next smallest part.
- Similar to Induction
- Dynamic programming optimally explores all the possible outcomes which makes it reliable and accurate.

Prisoner’s Dilemma

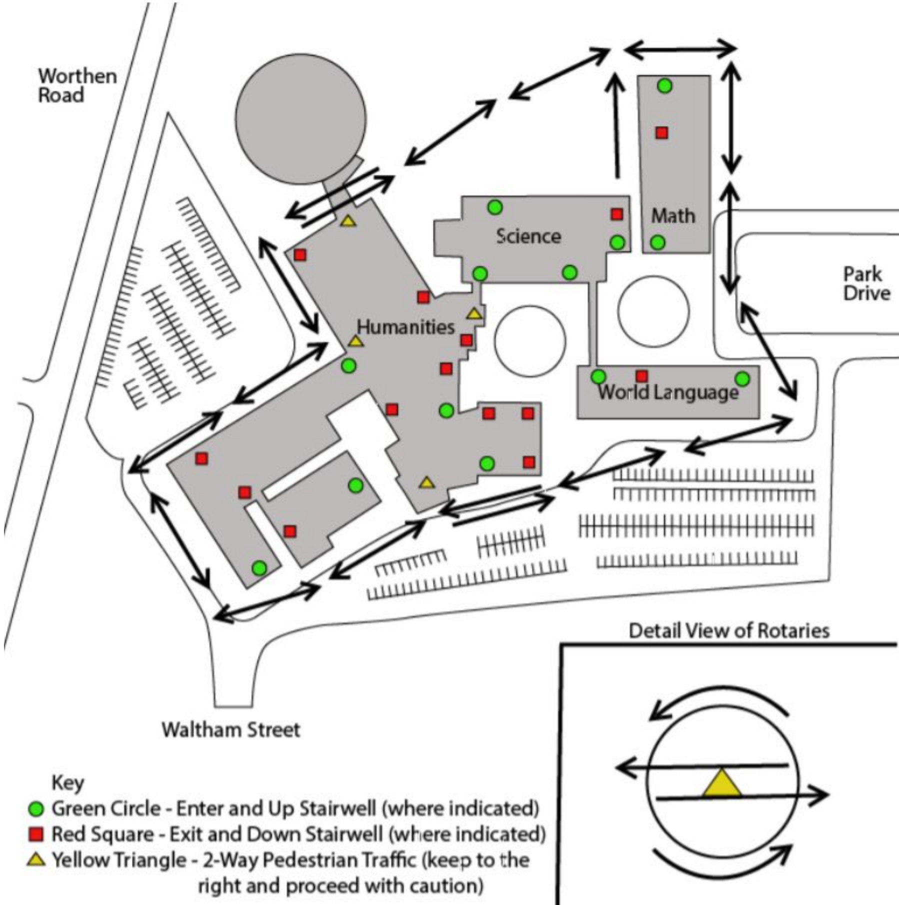
The dilemma here is in the perspective of each of the criminals betraying the other is in their best interest. However, this leads to them serving 2 years each instead of 1 year each.



	B stays silent (cooperates)	B betrays A (defects)
A stays silent (cooperates)	Both serve 1 year	A serves 3 years, B goes free
A betrays B (defects)	A goes free, B serves 3 years	Both serve 2 years

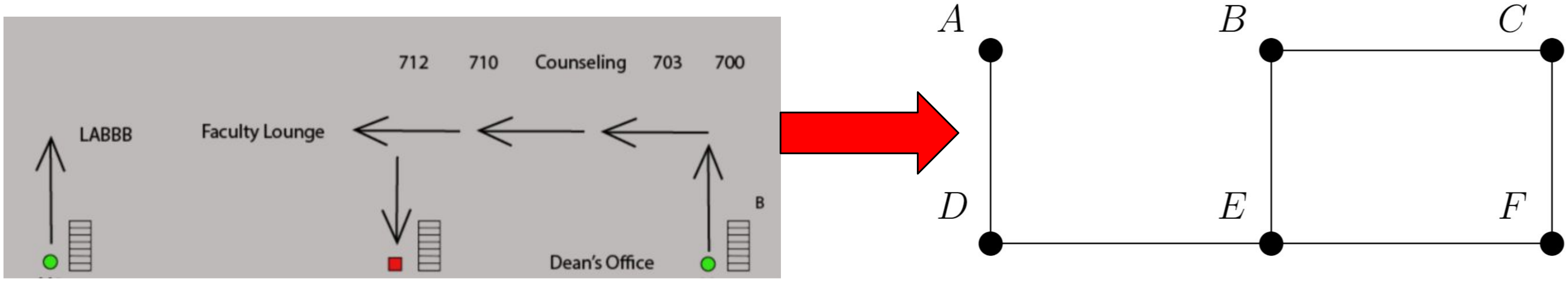
Applications of Combinatorics in Classroom Assignment Optimization

Problem statement: We want to find the best way for a group to traverse through the school taking the most efficient route. Where the total distance traveled by the group is the sum of the individual distances traveled by each student to reach their destination. We will use the building images of Lexington High School to create our graphs.


Worthen Road
Math
Science
Humanities
World Language
Park Drive
Waltham Street
Detail View of Rotaries
Key
🟢 Green Circle - Enter and Up Stairwell (where indicated)
🟥 Red Square - Exit and Down Stairwell (where indicated)
△ Yellow Triangle - 2-Way Pedestrian Traffic (keep to the right and proceed with caution)

School Mapping

We will use the Lexington High School Math Building Floor 1 and we can change the classrooms into points and the hallways into edges. For our purposes, we will assume that points A, B, C, D, E, F represent only classrooms and not classes and all hallways go in both directions.



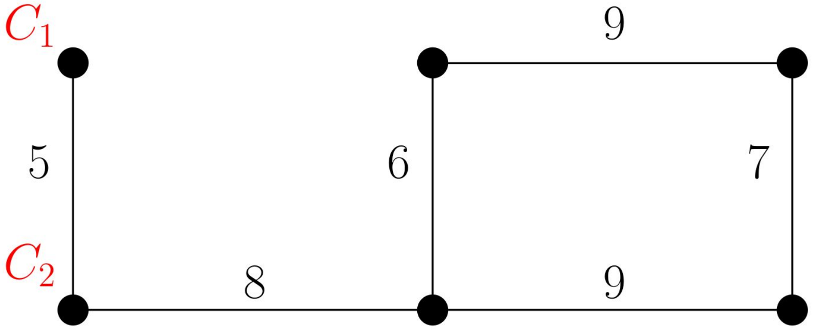
Greedy Algorithm Solution Example

Let us assume that there are 6 students $S_1, S_2, S_3, S_4, S_5, S_6$ that have 2 classes each and the distances between the edges are equal.

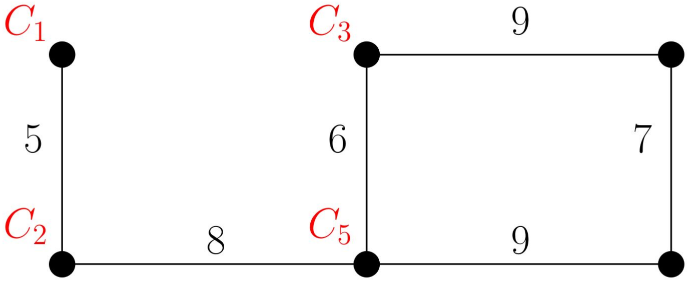
S_1 has classes C_1 and C_2
S_2 has classes C_3 and C_5
S_3 has classes C_4 and C_6
S_4 has classes C_2 and C_4
S_5 has classes C_6 and C_1
S_6 has classes C_3 and C_5

- Apply the Greedy Algorithm
- Minimize for Student 1
- Minimize for Student 2
- Continue through all students

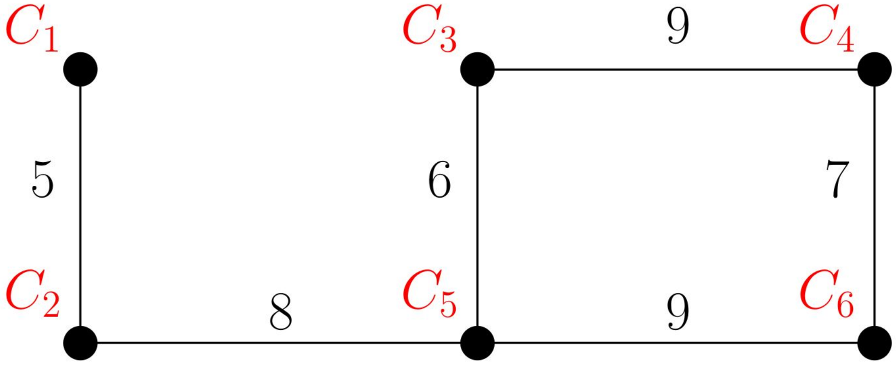
Graphs iterating through each student



Student 1



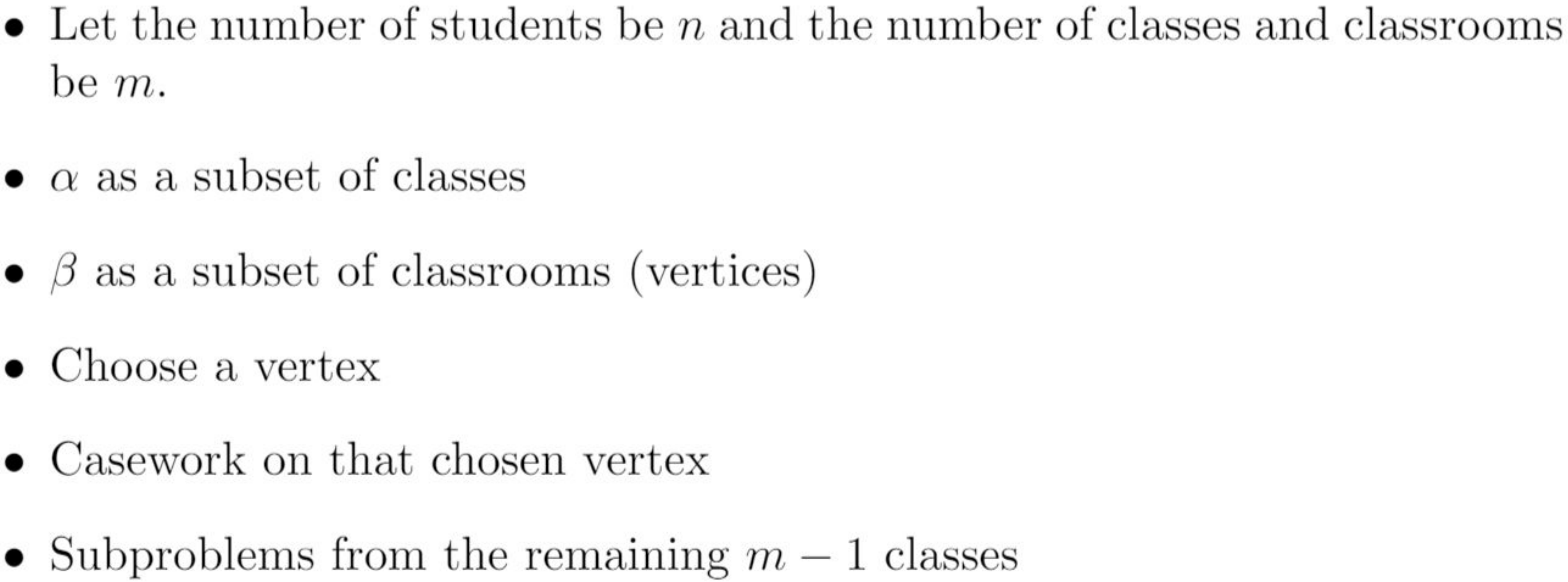
Student 1 and 2



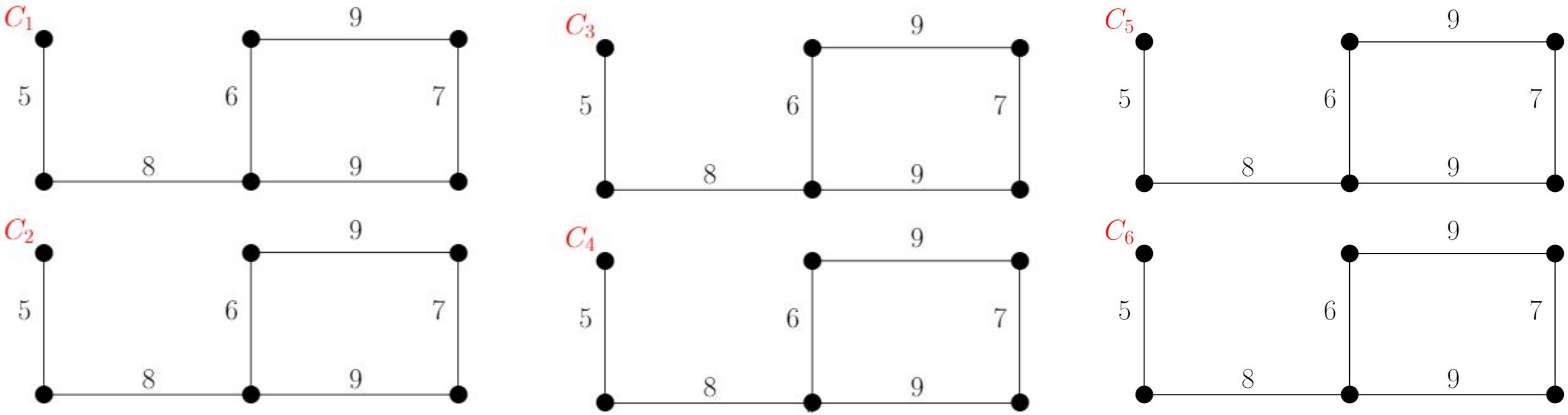
Student 1,2 and 3

Now that we have finished assigning the classrooms to all the points it does not matter what classes student 4,5,6 have as their classes overlap with classes that are already on the graph.

Dynamic Programming Solution Generalized



- Let the number of students be n and the number of classes and classrooms be m.
- α as a subset of classes
- β as a subset of classrooms (vertices)
- Choose a vertex
- Casework on that chosen vertex
- Subproblems from the remaining $m-1$ classes



Above we see how we would choose a point and do casework on that point by assigning it all 6 classes. Now we would have a subproblem with the remaining 5 classes that were not yet assigned.

Evaluation of Algorithms

Greedy Algorithm	Dynamic Programming
Optimizes by making a decision in the moment.	Optimizes by breaking down a problem using subproblems
Runs very quickly because it will iterate from each student and create the best scenario for that student	Runs more slowly because it breaks down the problem into smaller subproblems which does take significantly more time due to the numerous cases it has to execute.
Bias and it will give the best solutions to the first few students and those it meets last will not have as much choice	Fairness and Efficiency because it will go through all possible cases to find the best solution and all students will get an equal chance
Inaccurate because it misses many possibilities because it goes through the students 1 by 1 without ever thinking back to a student or thinking ahead for other students.	Accurate because it explores every possibility and gives you the best assignments for the class placement.

Acknowledgements

- PRIMES Circle for making this program and presentation possible
- Mentor: (Victor)Jung Soo Chu for guiding and teaching me throughout this semester
- Peter Haine for reading and editing my presentation
- Parents for their support throughout writing my presentation