# VIETNAM GENERAL CONFEDERATION OF LABOUR

# TON DUC THANG UNIVERSITY

# FACULTY OF INFORMATION TECHNOLOGY



# WEB PROGRAMMING AND APPLICATIONS

# COURSE ESSAY – WEB DEPLOYMENT AND APPLICATIONS

# *Supervisor:* MR. MAI VĂN MẠNH

*Author*: **NGUYỄN CHÂU THÙY LINH– 518H0031**

**NGUYỄN MINH NHỰT– 518H0545**

**DƯƠNG LÝ BẢO THÀNH– 518H0275**

**VÕ NGUYỄN MINH HUY– 518H0275**

Class: **18H50203–18H50204**

Course: **22**

**HO CHI MINH CITY, 2020**

# VIETNAM GENERAL CONFEDERATION OF LABOUR

# TON DUC THANG UNIVERSITY

# FACULTY OF INFORMATION TECHNOLOGY



# WEB PROGRAMMING AND APPLICATIONS

# COURSE ESSAY – WEB DEPLOYMENT AND APPLICATIONS

# *Supervisor:* MR. MAI VĂN MẠNH

*Author*: **NGUYỄN CHÂU THÙY LINH– 518H0031**

**NGUYỄN MINH NHỰT– 518H0545**

**DƯƠNG LÝ BẢO THÀNH– 518H0275**

**VÕ NGUYỄN MINH HUY– 518H0275**

Class: **18H50203–18H50204**

Course: **22**

**HO CHI MINH CITY, 2020**

Appreciation Letter

Firstly, this should be an honor to send our regards to the Faculty of Information Technology, lecturers and staff from all departments of Ton Duc Thang University. We would like to express our sincere thanks for the support and assistance during the implementation of the statistics and probability report.

We would like to express our gratitude to Mr. Mai Van Manh - teachers who directly instructed and supervised us to complete this essay.

We sincerely thank our friends and classmates who are studying and working at Ton Duc Thang University and the family has encourage, facilitated and helped us during the process.

Due to the fact that our actual ability is still weak, we ensure that we still have many shortcomings, so we hope our supervisor and the other professors will ignore it. At the same time, we hope to receive many comments from many sources to help us accumulate more experience to complete the upcoming graduation report to achieve better results.

THE ESSAY HAS BEEN CONDUCTED IN TON DUC THANG UNIVERSITY

We assure that this is our own product and has been guided by Mr. Mai Van Manh. The research contents, results in this topic are all about honesty. The data in the tables for analysis, comments and evaluation are collected by the us from various sources in the reference section.

In addition, comments and assessments as well as data from other authors or organizations are also used in the essay but with references and annotations.

**If there is any fraud is detected, we ensure our complete responsibility for the contents of our work.** Ton Duc Thang University is not related to violations of authority and copyright caused by us during our work process (if any).

*Ho Chi Minh City, Sunday, 3rdMay. 2019*

*Author*

*(Sign and provide full name)*

*Dương Lý Bảo Thành Nguyễn Châu Thùy Linh*



*Võ Nguyễn Minh Huy Nguyễn Minh Nhựt*

VERIFICATION AND EVALUATION FROM LECTURER

**Supervisor’s evaluation**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ho Chi Minh city, date:

(Sign and provide full name)

**Marking lecturer’s evaluation**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ho Chi Minh city, date:

(Sign and provide full name)

SUMMARY

In this essay, I am going to satisfy report the study with 6 chapters below:

**Chapter 1:** Social Networks and Social network analysis problems

**Chapter 2:** Social Network measurements

**Chapter 3:** Community Detection

**Chapter 4:** Demo

**Chapter 5:** Conclusion and Discussions

**Chapter 6:** Self – evaluation

**Chapter 7:** References

TABLE OF CONTENTS

[VIETNAM GENERAL CONFEDERATION OF LABOUR 1](#_Toc24233931)

[TON DUC THANG UNIVERSITY 1](#_Toc24233932)

[FACULTY OF INFORMATION TECHNOLOGY 1](#_Toc24233933)

[DISCRETE STRUCTURES ESSAY 1](#_Toc24233934)

[Supervisor: MR.TRAN HONG TAI 1](#_Toc24233935)

[Author:NGUYENMINHNHUT– 518H0545 1](#_Toc24233936)

[VIETNAM GENERAL CONFEDERATION OF LABOUR 2](#_Toc24233937)

[DISCRETE STRUCTURES ESSAY 2](#_Toc24233938)

[Supervisor: MR.TRAN HONG TAI 2](#_Toc24233939)

[Appreciation Letter i](#_Toc24233940)

[VERIFICATION AND EVALUATION FROM LECTURER iii](#_Toc24233941)

[Supervisor’s evaluation iii](#_Toc24233942)

[Marking lecturer’s evaluation iii](#_Toc24233943)

[SUMMARY iv](#_Toc24233944)

[TABLE OF CONTENTS 1](#_Toc24233945)

[LIST OFTABLES AND ILLUSTRATIONS 4](#_Toc24233946)

[LIST OF ILLUSTRATION 4](#_Toc24233947)

[LIST OF TABLES 4](#_Toc24233948)

[CHAPTER 1- GROUP INTRODUCTION 5](#_Toc24233949)

[1.1 Overview: 5](#_Toc24233950)

[1.2 List of my group members: 5](#_Toc24233951)

[*Table1.1: Personal information of group member* 5](#_Toc24233952)

[1.3My each week work: 5](#_Toc24233953)

[*Table 1.2: Details on weekly work* 5](#_Toc24233954)

[1.4Introduction to other chapters: 6](#_Toc24233955)

[CHAPTER 2 – REVERSE POLISH NOTATION AND TRUTHTABLE APPLICATION 7](#_Toc24233956)

[2.1 Overview 7](#_Toc24233957)

[2.2 What is Reverse Polish Notation (RPN)? 7](#_Toc24233958)

[2.3 How to actually execute RPN? 7](#_Toc24233959)

[*2.3.1 Infix to Postfix Conversion:* 8](#_Toc24233960)

[Table 2.1: Postfix conversion example 9](#_Toc24233961)

[*2.3.2 Postfix Evaluation:* 9](#_Toc24233962)

[2.4 Basic logic for Truth-table application 10](#_Toc24233963)

[*Table 2.2: Table of precedence* 10](#_Toc24233964)

[*Table 2.3: Propositional logic postfix conversion example* 11](#_Toc24233965)

[*Picture 2.1: Binary table* 12](#_Toc24233966)

[*Picture 2.2: Operands table with 3 operands* 12](#_Toc24233967)

[CHAPTER 3: PROGRAM EXPLANATION 14](#_Toc24233968)

[3.1 Codes section: 14](#_Toc24233969)

[*Picture 3.1: Code screenshot#1: Stack simulation* 14](#_Toc24233970)

[*Picture 3.2: Code screenshot #2: def Infix2Postfix* 15](#_Toc24233971)

[*Picture 3.3: Code screenshot#3: def Postfix2Truthtable* 16](#_Toc24233972)

[*Picture 3.3.1: Code screenshot#3: def Postfix2Truthtable (continue)* 17](#_Toc24233973)

[3.2 Details explanation: 17](#_Toc24233974)

[Picture 3.4: Two first test-cases 18](#_Toc24233975)

[3.2.1 Code explanation in detail: 18](#_Toc24233976)

[Picture 3.5: Inherence of Stack and other variables 18](#_Toc24233977)

[Picture 3.6: Activity inside Stack and road to postfix 19](#_Toc24233978)

[19](#_Toc24233979)

[Picture 3.7: Checking priority of each token if it is an operator 19](#_Toc24233980)

[Picture 3.8: Returning Postfix 20](#_Toc24233981)

[Picture 3.9: define a function named implies to do implication 20](#_Toc24233982)

[Since a truth-table based mostly on its operands so I create a list to store only operands 20](#_Toc24233983)

[Picture 3.10: Operands getter and calculating numbers of rows using number of operands 20](#_Toc24233984)

[Picture 3.11: Processing my sub-table 21](#_Toc24233985)

[Picture 3.12: Tuple up my sub list 21](#_Toc24233986)

[Picture 3.13: Calculating each token 22](#_Toc24233987)

[Picture 3.14: Final step 23](#_Toc24233988)

[3.2.2 Manual running with test-cases: 23](#_Toc24233989)

[CHAPTER 4: EXPERIMENTAL RESULTS 26](#_Toc24233990)

[*Picture 4.1: Screenshot test-case 1* 26](#_Toc24233991)

[*Picture 4.2: Screenshot test-case 2* 27](#_Toc24233992)

[*Picture 4.3: Screenshot test-case 3* 27](#_Toc24233993)

[*Picture 4.4: Screenshot test-case 4* 28](#_Toc24233994)

[*Picture 4.5: Screenshot test-case 5* 29](#_Toc24233995)

[CHAPTER 5: REFERENCES 30](#_Toc24233996)

LIST OFTABLES AND ILLUSTRATIONS

**LIST OF ILLUSTRATION**

[*Picture 2.1: Binary table* 12](#_Toc24233966)

[*Picture 2.2: Operands table with 3 operands* 12](#_Toc24233967)

[*Picture 3.1: Code screenshot#1: Stack simulation* 14](#_Toc24233970)

[*Picture 3.2: Code screenshot #2: def Infix2Postfix* 15](#_Toc24233971)

[*Picture 3.3: Code screenshot#3: def Postfix2Truthtable* 16](#_Toc24233972)

[*Picture 3.3.1: Code screenshot#3: def Postfix2Truthtable (continue)* 17](#_Toc24233973)

[Picture 3.4: Two first test-cases 18](#_Toc24233975)

[Picture 3.5: Inherence of Stack and other variables 18](#_Toc24233977)

[Picture 3.6: Activity inside Stack and road to postfix 19](#_Toc24233978)

[Picture 3.7: Checking priority of each token if it is an operator 19](#_Toc24233980)

[Picture 3.8: Returning Postfix 20](#_Toc24233981)

[Picture 3.9: define a function named implies to do implication 20](#_Toc24233982)

[Picture 3.10: Operands getter and calculating numbers of rows using number of operands 20](#_Toc24233984)

[Picture 3.11: Processing my sub-table 21](#_Toc24233985)

[Picture 3.12: Tuple up my sub list 21](#_Toc24233986)

[Picture 3.13: Calculating each token 22](#_Toc24233987)

[Picture 3.14: Final step 23](#_Toc24233988)

[*Picture 4.1: Screenshot test-case 1* 26](#_Toc24233991)

[*Picture 4.2: Screenshot test-case 2* 27](#_Toc24233992)

[*Picture 4.3: Screenshot test-case 3* 27](#_Toc24233993)

[*Picture 4.4: Screenshot test-case 4* 28](#_Toc24233994)

[*Picture 4.5: Screenshot test-case 5* 29](#_Toc24233995)

**LIST OF TABLES**

[*Table1.1: Personal information of group member* 5](#_Toc24233952)

[*Table 1.2: Details on weekly work* 5](#_Toc24233954)

[Table 2.1: Postfix conversion example 9](#_Toc24233961)

[*Table 2.2: Table of precedence* 10](#_Toc24233964)

[*Table 2.3: Propositional logic postfix conversion example* 11](#_Toc24233965)

# CHAPTER 1- SOCIAL NETWORKS AND SOCIAL NETWORK ANALYSIS PROBLEMS

## 1.1 Overview:

In this section, I am going to present those details below:

* 1. *Introducing the concept of the social network*
  2. *The importance of social network*
  3. *Graph applications in social network modeling and how to model it using a graph*

## 1.2 Introducing to the concept of the social network:

Back in the days, people had done so many ways to communicate and keep that close contact with their relatives, friends and more; some of them can be pigeon post, handwritten letter, and telegraph, etc. Nowadays, it is so easy for people to keep in touch and maintain their relationship, thanks to the invention of phone and more than that, the internet.

The concept that we use Social network is kind of undefined but somehow we still can gather them as that we shall call them “The nature of Sociology”, different aspect but with the same attitude,

## 1.3 The importance of social network:

It has been a long time since the very first social network site launched, such as SIX DEGREES (1997 – 23 years ago). Then we cannot deny how social network changed the way we communicate everyday of our lives, it can even be our daily basis

### 1.3.1 The importance of social network in every life:

Social Network has become one of the most important parts of our lives as it allows us to communicate with many people, people that we know and people that we have not known yet too.

As we are living everyday of our lives, we always do these activities: Capturing, preserving and sharing.

Back in the day, it was that much difficult to capture things or moments due to the lack of devices, which was able to support that type of activity, and it was even more difficult to preserve the products that you had just made after capturing that we had to keep them in album or some sorts of books. None of these above, sharing in the past means even more than that cause we had so less ways to make it work the right way but the privacy since someone still can afford the album, books or things in your house.

Nowadays, thanks to the invention of social network, it makes everything much simpler to execute all those three activities. We now can access to nearly everything just with one click. Let think about this situation, your family and some of your relatives just went out for nice picnic but at the end, you were the only person with the picture of the rest, so what are you going to do then? Going to the photo shop and make a copies of the picture and pay for them then go and send the copies to everyone? The answer is “no”, all the things you have to do is sharing them on the site of any social network that you like such as Facebook or Twitter, etc. Suddenly, everyone with a social network account can have a copy.

Somehow, the social network sticks to your life as an integral part that you cannot deny, every morning of yours, what are you going to do next with that curiosity in your head wanting to know more about this world. Definitely, turning on the phone and open that web page then read the E-newspaper and I compromise that would be the first thing you will do in the morning.

### 1.3.2 The importance of social network in business:

By discussing the importance of social network in everyday life of a person, we by this way or the other ways still can come up with how social network plays a major role in business life.

Obviously, it is about to gain the contacts, clients and public awareness of the image of business. Even if you are an entrepreneur and you are running a small business or your business is managed to work from home, you are still in advantage, which means the resource you are holding is about to set up a global presence.

## 1.4 Graph applications in social network modeling:

In this section, I will focus on explaining how graph can be used to improve the performance of social network modeling by answering these questions below:

* What is a graph?
* What is social networking?
* What are applications of graph in social network and how to model it using a graph?
* Some problems in social network analysis

### 1.4.1 What is a graph?

According to book named Graph theory with application by C.Vasudev,  *a* graph G consists of a set of objects V = {v1, v2, v3,…} called vertices (or points, nodes) and other set E = {e1, e2, e3,…} whose elements are called edges(or lines, arcs).

The set V (G) called vertex set of G, E (G) is the edge set, and a graph usually denoted as

G = (V, E).

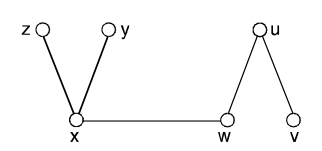
Let G be a graph and {u, v} an edge of G. Since {u, v} is 2-element set, we may write {v, u} instead of {u, v}. It is often more convenient to represent this edge by uv or vu.

If *e = uv* is an edge of a graph G, then we say that *u* and *v* are adjacent in G and that *e* joins u and v. (We may say that each that of u and v is adjacent to or with the other).

For example: A graph G is defined by the sets

V (G) = {u, v, w, x, y, z} and E(G) = {uv, uw, wx, xy, xz}.

Let consider the graph below, we have:



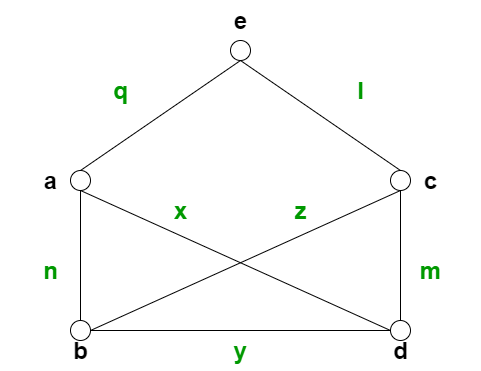
#### Picture 1.1: Example graph 1

Every graph has a diagram associated with it. The vertex u and an edge e are incident with each other as are v and e. If two distinct edges say e and f are incident with a common vertex, then they are adjacent edges.

A graph with p-vertices and q-edges is called a (p, q) graph.

The (1, 0) graph is called trivial graph. In the following figure the vertices a and b are adjacent but a and c are not.

The edges x and y are adjacent but x and z are not. Although the edges x and z intersect in the diagram, their intersection is not a vertex of the graph.

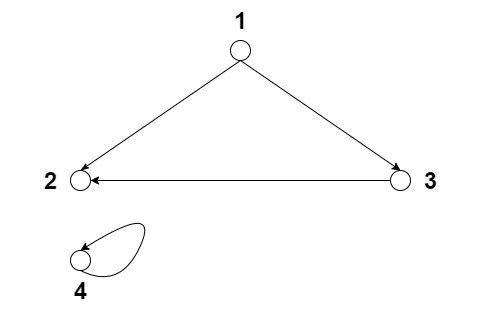


#### Picture 1.2: Example graph 2

For example:

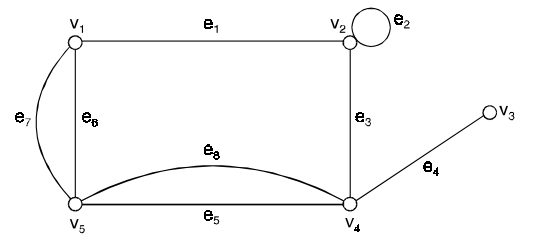
* Let V = {1, 2, 3, 4} and E = {{1, 2}, {1, 3}, {3, 2}. {4, 4}}.

Then G(V, E) is a graph



#### Picture 1.3: Example graph 3

* Let V = {1, 2, 3, 4} and E = {{1, 5}, {2, 3}}. Then G(V, E) is not a graph, as 5 is not in V.
* A graph with 5-vertices and 8-edges is called a (5, 8) graph.



#### Picture 1.4: Example graph 4

### 1.4.2 What is social networking?

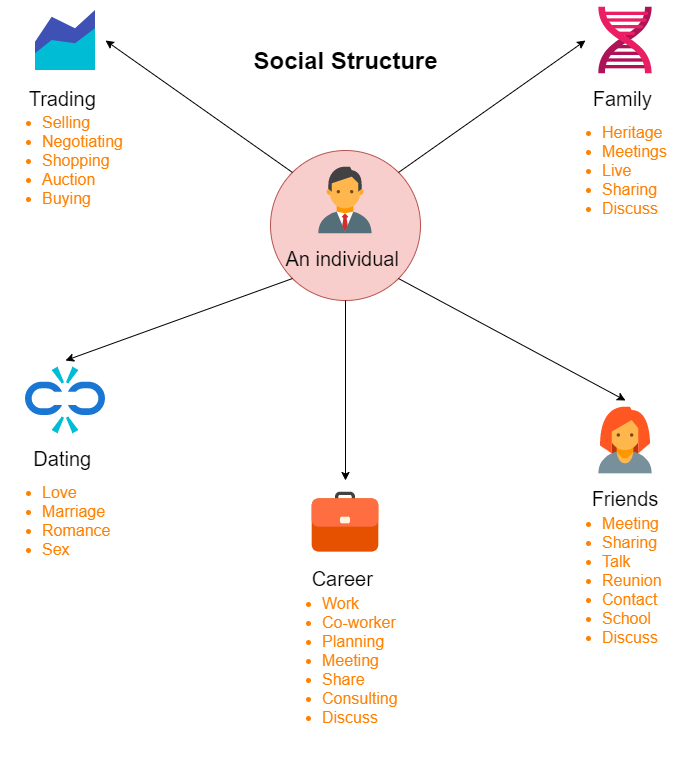
According to *Investopedia*, Social networking is use of internet – based social media to stay connected with other people. People use social network with different purposes such as social connection, business purposes such as increasing recognition or brand loyalty.

Consider that social network cam be either online and offline. Since social network is a network of personal correlations of community – based interaction, an online social network is a web app or site that validates users exchange information on the internet by sharing multiple types of data such as: images, videos, files, texts, etc.



#### Picture 1.5: Online social network as a web-app or site

We can give out the definition of a social network as a social structure, which we have spent our daily lives living in it. Denotation of a social network comes naturally and we do not need to do any research to get to know about it.



#### Picture 1.6: Social Network as a daily Social Structure (Society)

In conclusion, social network is an important application of World Wide Web, it makes us capable to communicate and keep the contact with people. It is difficult to deny how social network changed our society and the way we look at current life, and it plays an integral part in assist every single task in our daily lives.

### 1.4.3 What are applications of graph theory in social network analysis and how to model it using a graph?

Nevertheless, why graph is an application in social network? According to Dawn Iacobucci, graph theory has been useful for many reasons.

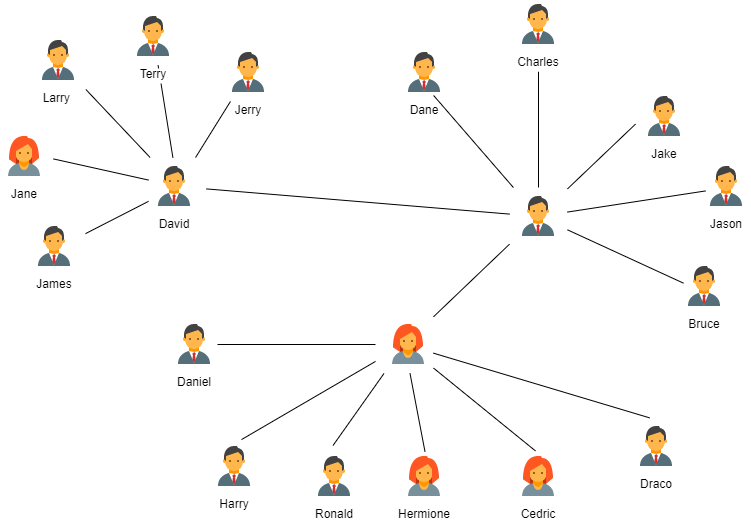
Firstly, graph theory provides a vocabulary, which can be used to label and denote many social structural properties. This vocabulary also gives us a set of primitive concepts that allows us to refer quite precisely to these properties.

Secondly, graph theory gives us mathematical operations and ideas with which many of these properties can be quantified and measured.

Finally, given this vocabulary and these mathematics, graph theory gives us the ability to prove theorems about graphs, and hence, about representations of social structure. Like other branches of mathematics, graph theory allows researchers to prove theorems and deduce testable statements.

However, as Barnes and Harary (1983) have noted, "Network analysts ... make too little use of the theory of graphs". Although the representation of a graph and the vocabulary of graph theory are widely used by social network researchers, the theorems and derivations of graph theory are less widely used by network methodologists.

But how to model a social network using a graph? Here is an example for the question how. A Social network is featured as a scalar graph G = (V, E) where V is denoted as a set of vertices (node), E is a set of edges (link, relation), which means instead of naming nodes with alphabetic based symbols; we turn them into people representation.



#### Picture 1.7: Graph representation of a social network

## 1.5 Some problems in social network analysis

After reading some online materials and books, there are some pieces that are vague, unclear due to their low level in persuasion but I do believe there are 2 biggest problems at heart of social network analysis and the answer is yes that they are left unsolved and being under investigation.

First, it is unable to generate a social network artificially that can be looked as a one in real life. Obviously, some progresses can present the possibility:

* The preferential attachment gets much closer to a real social network more than a graph model did
* Assortativity could be another big step closer to reality

Overall, relationship in human world is not just difficult to re-generate but also complicated to put all of the things together.

Second, online behavior is just a subset of real life behavior, which means the assumption’s fragility of our analysis is somehow still high and the result is damageable.

# CHAPTER 2 – SOCIAL NETWORK MEASUREMENTS

## 2.1 Overview

In this chapter, a research about representation of a social network by using graph will be performed with the explanation of some concepts and measurements of the following kind of graph

*2.2 How to represent a social network by using graph in detail*

*2.3 An existed social network as an example*

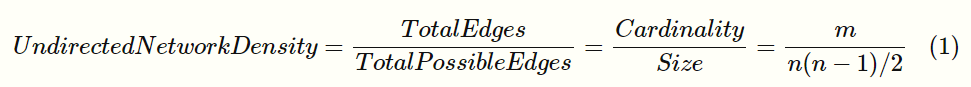
## 2.2 How to represent a social network by using graph in detail

As I have mentioned above, we should use graph to represent a social network in which nodes are actors (users), edges are connections or relationships among users. Before going to details, we need to research definition of those below:

* Network density, degree centrality, closeness centrality, betweenness centrality, clustering centrality
* Key players and how to detect key players

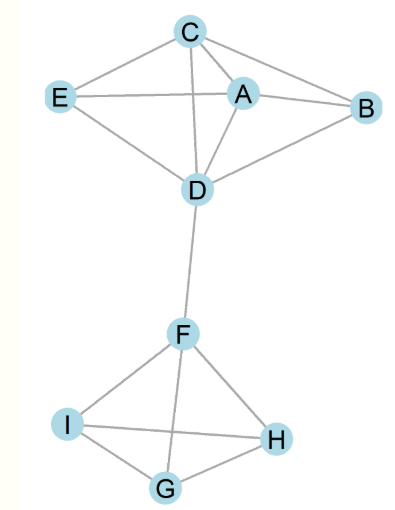
### Nework density

Density is used to address the communal connections among these nodes and addresses the quality of interpersonal relations. The more mutual cross-linkages there are, the more likely a person can get resources and help. According to Han Lin and Lin Qui, they mentioned the egocentric network size and density using the social network analysis software UCINET. The network size for each participant is the total number of friends in his or her friend list (M =507.49, SD = 241.79). The network density is the ratio of existing connections among friends over all possible connections in the 1-step egocentric network (M = 0.05, SD = 0.03). As we many know, there are 2 types of graph – directed and undirected graph then there are 2 different formula to calculate network density for each type, first, let move to undirected graph density calculus:



#### Formula 2.1: Undirected graph density calculus

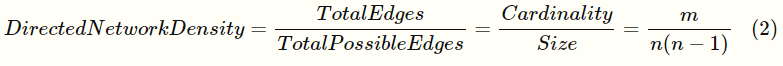
For example, consider an undirected graph below:



#### Picture 2.1: Undirected graph sample

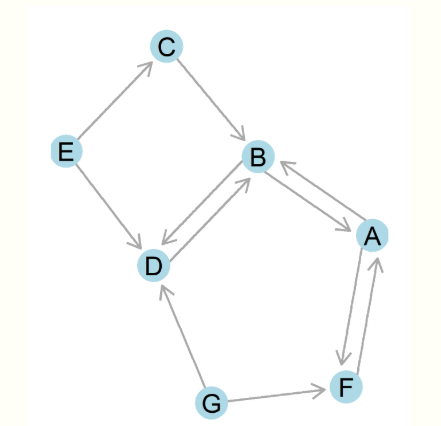
There are 9 nodes in our graph so there must be (9\*8)/2 or 36 possible edges. Thus, the density must be 0.44

How about directed graph? Density calculus should be this one below:



#### Formula 2.2: Directed graph density calculus

For example, consider a directed graph below:



#### Picture 2.2: Directed graph sample

Applying the given formula, we can simply find the density is 11/(7\*6), or 0.26.

Before moving to Network centrality, I need to get everything to the point

### Degree centrality

Defining the degree centrality is defining who is the center of the rest of the social network, who holds important, advantaged position in the network but there are many ways to reach there. By the shortest way, we need to find

## 2.3 How to actually execute RPN?

Briefly, we have to do these steps:

* Turn our infix into a postfix.
* Evaluating the postfix to get the final result

### *2.3.1 Infix to Postfix Conversion:*

Firstly, we have to turn a given into a postfix and how to do that? Everything is wrapped up into one simple line, which is: “Pop it out when it is greater or equal in priority”, but in detail, there is complete Infix to postfix conversion algorithm that uses one thing called “stack” to hold the operators rather than operands.

Step by step, we shall obey to those instructions below:

1. Print operands as they arrive.

2. If the stack remains empty or contains a left parenthesis on top, push the incoming operator onto the stack.

3. If the incoming symbol is a left parenthesis, push it on the stack.

4. If the incoming symbol is a right parenthesis, pop all operators from the stack and print them until a left parenthesis is met. Then discard the pair of parentheses.

5. If the incoming symbol has higher precedence than the top of the stack, push it on the stack.

6. If the incoming symbol has equal precedence with the top of the stack, use association. If the association is left to right, pop and print the top of the stack and then push the incoming operator.

7. If the incoming symbol has lower precedence than the symbol on the top of the stack, pop the stack and print the top operator. Then test the incoming operator against the new top of stack.

8. At the end of the expression, pop and print all operators on the stack. (No parentheses should remain).

But, to give you a brighter corner of the job, I will demonstratethe algorithm by some examples, for instance, our infix is 1\*3^2+2, as we solve it naturally, the result should be 11 where the exponentiation and multiplication must be finished before addition, more specific:



#### Table 2.1: Postfix conversion example

### *2.3.2 Postfix Evaluation:*

After receiving the postfix, we should move on by finishing the rest of the algorithm with evaluating that postfix with postfix evaluation algorithm. To make the algorithm less complex, I turn it in some lines of sudo codes:

for each token in the postfix expression:

if token is an operator:

operand\_2 ← pop from the stack

operand\_1 ← pop from the stack

result ← evaluate token with operand\_1 and operand\_2

push result back onto the stack

else if token is an operand:

push token onto the stack

result ← pop from the stack

Since this point, we can see how a processor can be written to come up with such a result for every natural calculation we enter, it usually gets through with 2 algorithms above – Infix to Postfix conversion and Postfix Evaluation. So, how postfix evaluation works, below is an illustration:

**1 3 2 ^ \* 2 +** =

1**3**2^\*2+ =

13**2**^\*2+ =

132**^**\*2+ =

19\*2 + =

9**2**+ =

9 2**+** =

11 =

**11**

## 2.4 Basic logic for Truth-table application

In section 2.2 and 2.3 above, I presented Reverse Polish Notation (RPN) on natural numbers or our operands here are numbers only but in proportional logic, we usually research by using alphabets so it is obvious that we should turn our operand type into alphabetic form but can our RPN still recognize them? And the answer is ‘Yes’, there will be no problem with that. But how we execute it with this kind of infix here: (P|Q&R)>K? Apparently, type of operator is also changed and there will be no addition, multiplication… etc. And they are replaced with specific logic operators such as and (&), or (|), implication (>) or bi-implication (=). By that operator replacement we have a different sort of technique to solve it but the idea for the rest still remains the same, there are just some differences inside the algorithm.

Instead of converting by switching the mathematical operators to the back of their numeric operands, we now set the logic operand to the back with the same technique used above in section 2.3 but with different precedence for logic calculating symbols, more specifically, our precedence will follow the table below:

|  |  |  |
| --- | --- | --- |
| Operator | Meaning | Precedence |
| ~ | Negation | 1 |
| & | And | 2 |
| | | Or | 3 |
| > | Implication | 4 |
| = | Bi-Implication | 5 |

### *Table 2.2: Table of precedence*

According to table 2.2 and above Infix-to-Postfix Conversion, we can convert a propositional Logic infix into a postfix, for example, our infix is R|(P&Q) then we have step-by-step-table below:

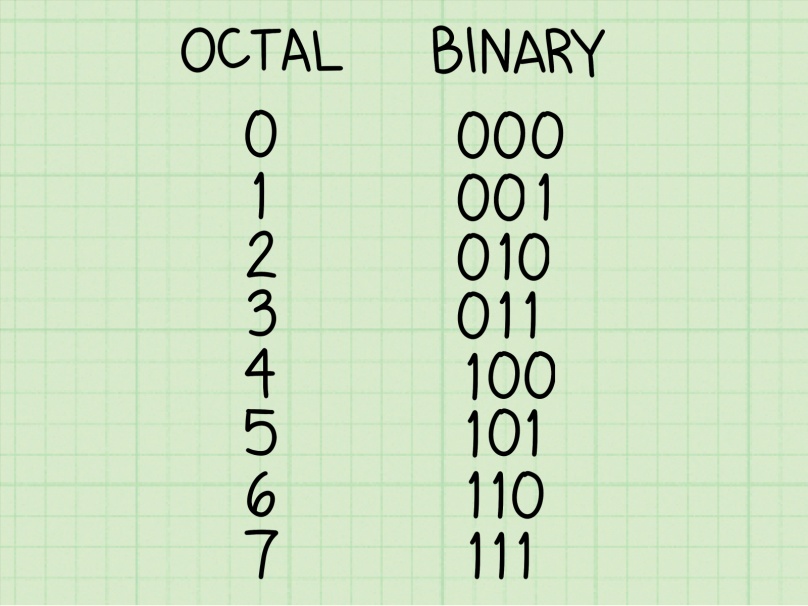
|  |  |  |  |
| --- | --- | --- | --- |
| Step | Current operand | Stack | Postfix string |
| 1 | R |  | R |
| 2 | | | | | R |
| 3 | ( | | ( | R |
| 4 | P | | ( | R P |
| 5 | & | | ( & | R P |
| 6 | Q | | ( & | R P Q |
| 7 | ) | | | R P Q & |
| 8 |  |  | R P Q & | |

### *Table 2.3: Propositional logic postfix conversion example*

So, our postfix should be R P Q & | if I am correct, next step should be evaluating postfix but with logic postfix we cannot print out number or another logical clause as the result so we have to use that postfix and generate a truth-table to present all statuses inside the logical clause, so how we can do that? Since we know the basic of postfix evaluation technique above and wrapping every 2 operands and 1 operator then calculate them, we shall use the same procedure for logic postfix, every 2 logic operand and 1 logic operator so we can render a column of truth-table instead of pushing a number as the result back to the stack then keep on with the others.

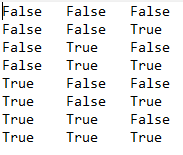
Saying is much simpler than doing so let me be brief, with my codes which I will put them in Chapter 3 below, I did use the same technique as I presented above but with some changes in format. My main idea is to divide the truth-table into 2 parts: Operands Only Table is first part and Logical Calculation is second one.

First part is Operands Only Table, thanks to a helpful guidance from Mr.  
Shad Sluiter from Grand Canyon University, Icould come up with solution for this part by using binary code to represent truth-table, as we many know that binary forms of numbers are actually the order and a face of truth-table elements:



### *Picture 2.1: Binary table*

The picture above shows binary form of numbers from 0 to 7, if we replace 0 with False and 1 with True we will definitely extract a complete truth-table for logical expression with 3 operands because with 3 operands we shall have 23 = 8 rows so the loop which runs from 0 to 7 may extract enough rows and nodes for that expression:



### *Picture 2.2: Operands table with 3 operands*

Our first part is finished, now is second one which is Logical postfix evaluation and it is just the same as postfix evaluation which we used above to reach final result for numeric or alphabetic postfix but the evaluation we are going to use is a bit different since the variable “result” will be replaced with a list of rows for truth-table.

Array\_contains\_operand = []

Array\_contains\_result = []

for each token in the postfix expression:

if token is an operator:

Array\_contains\_operand ← token

if token is an operator:

operand\_2 ← pop from Array\_contains\_operand

operand\_1 ← pop from Array\_contains\_operand

Array\_contains\_result ← evaluate token with operand\_1 and operand\_2

ListA ← Tuple of Operands Only

ListB ← Zip every number\_of\_operators element inside Array\_contains\_result

Return ListA + ListB

Sudo codes above shows my very first idea for generating a truth-table from a postfix where the key is number of columns in ListB is actually number of operators that appear in Postfix. Since that point of recognition, there is no doubt for our truth-table, which means, it is right or wrong depends on out postfix. To solve that problem, I shall substantiate all in my Python program in Chapter 3 and testing the result in Chapter 4.

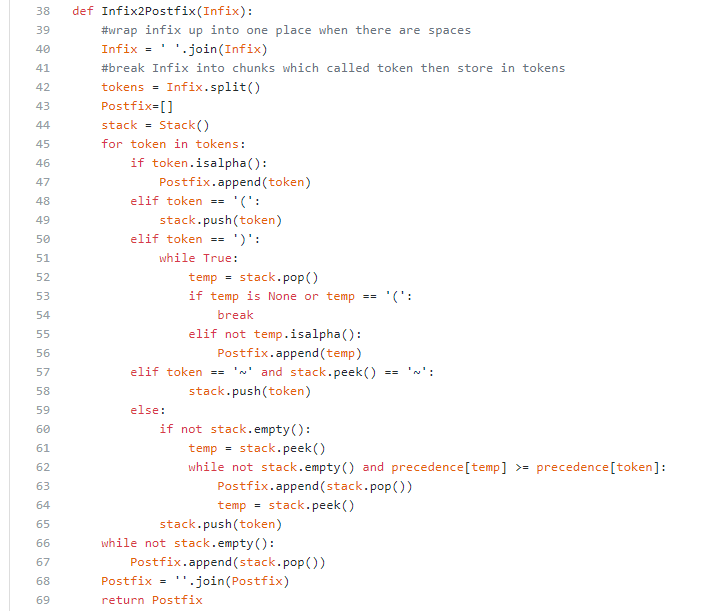
# CHAPTER 3: PROGRAM EXPLANATION

In this chapter, my programming part will be shown and the other parts will be hidden to make sure that I can deliver the briefest explanation on the codes.

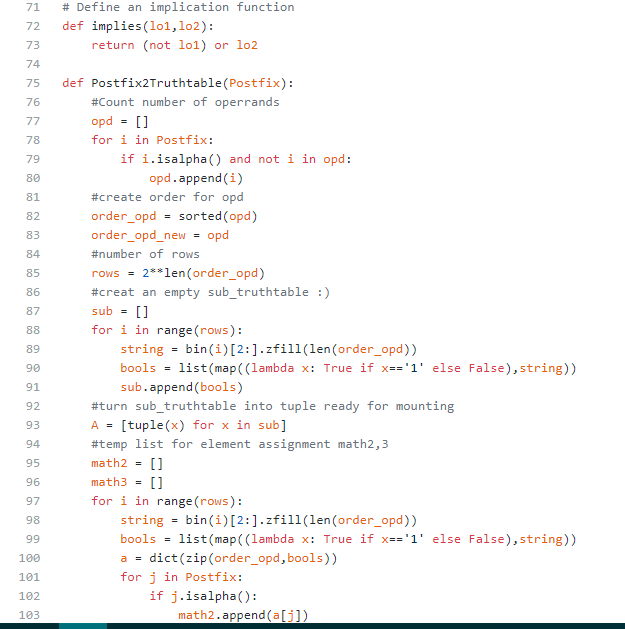
## 3.1 Codes section:

******

### *Picture 3.1: Code screenshot#1: Stack simulation*



### *Picture 3.2: Code screenshot #2: def Infix2Postfix*



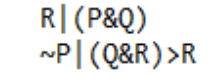
### *Picture 3.3: Code screenshot#3: def Postfix2Truthtable*



### *Picture 3.3.1: Code screenshot#3: def Postfix2Truthtable (continue)*

## 3.2 Details explanation:

In this section, I am going to explain my 2 functions *Infix2Postfix* and *Postfix2Truthtable* in detail and I will check my program with 2 first test-cases manually:

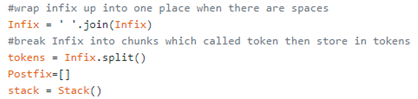


#### Picture 3.4: Two first test-cases

### 3.2.1 Code explanation in detail:

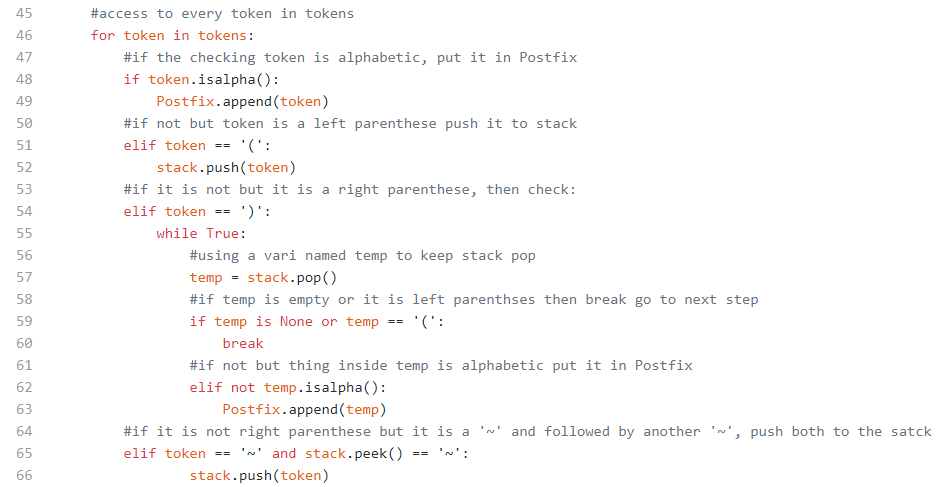
Firstly, I should start over with creating a class Stack to implement all function of a stack to my program, since Python does not have a class stack or if it does, we have to install it.

Then, we move to *Infix2Postfix* function:



#### Picture 3.5: Inherence of Stack and other variables

Then, we access to all the tokens in Infix string to check, I go from the parentheses to comparing precedence of every operators in the Infix, by that we will not miss anything

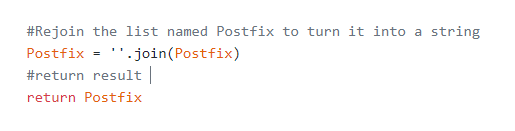


#### Picture 3.6: Activity inside Stack and road to postfix

### 

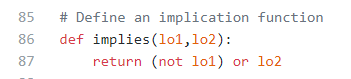
#### Picture 3.7: Checking priority of each token if it is an operator

Then, I rejoin all elements in one place using function ‘join’ to join each token together so we can receive such a postfix string:



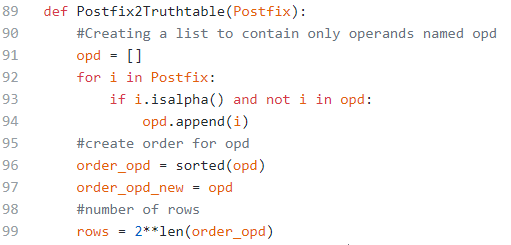
#### Picture 3.8: Returning Postfix

After that we, deal with the truth-table by using function *Postfix2Truthtable.* First of all, I need to implement all of the things I need, one of them is implication and as we claim it P -> Q is equal with ~P | Q so we have function below:



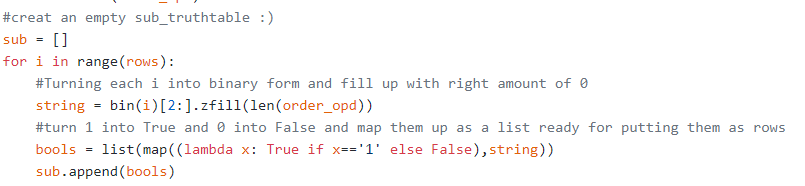
#### Picture 3.9: define a function named implies to do implication

### Since a truth-table based mostly on its operands so I create a list to store only operands



#### Picture 3.10: Operands getter and calculating numbers of rows using number of operands

As I stated above, a truth-table has 2 parts in it so I have to create a sub-table contains only operands so it will be like this:



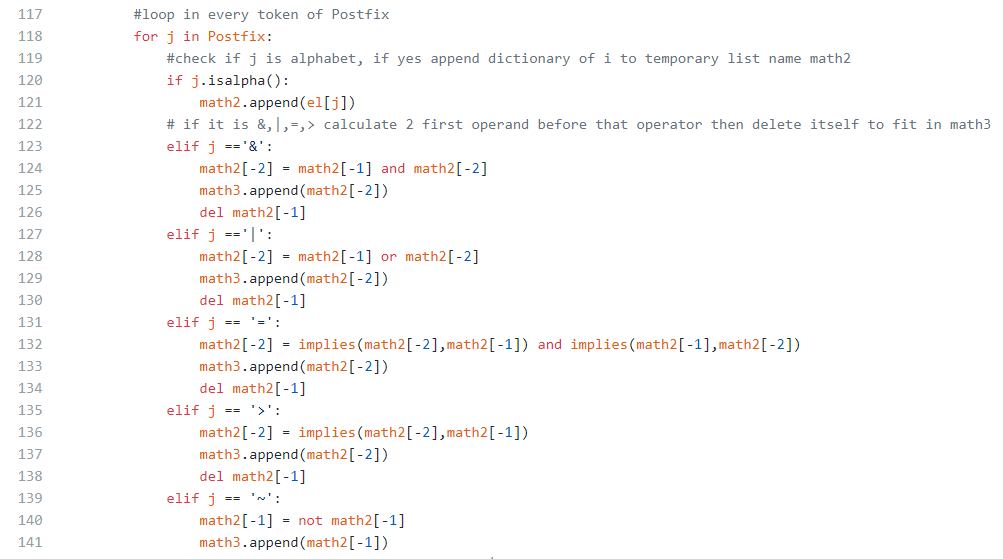
#### Picture 3.11: Processing my sub-table

In this step, most of readers will question themselves, why there happens so many odd things like string, bools , bin stuff and list(map), etc. But it is just way to render my truth-table because I use binary code to generate my truth-table so there definitely is a bin function, which the for loop will be responsible for loop from 0 to number of rows which is 2number of operands so it will turn these number into binary code but in Python number 0 will be reduced by every stage I have to use zfill function which help me to fill up number of ‘0’ to right amount of operands that is why I have ‘zfill(len(order\_opd))’, I did not have to use order\_opd in this case because with this technique I will not have to sort my list to alphabetic order but by putting it in, I can easily manage my codes which is my own coding habit. Then I map them up by using map(function) and putting them into a list so I have a list of True, False thing from my binary render, after that I append all of list in 1 list which is sub – my sub-table. But it does not stop here, my list does not have a form of a tuple since it is still a combination of lists so I have to tuple-up all of them one more time



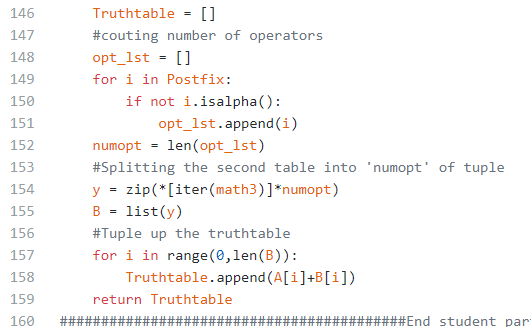
#### Picture 3.12: Tuple up my sub list

At this stage A is an official name for my operand-only list. Next up, I will deal with list of calculation where there are only logic calculating matter columns. With this type of problem, I will use the same technique above which was used for my operand-only list but with an advance, I will turn my Boolean map into a dictionary with order as {‘P’: False, ‘Q’: False, ‘R’: False}, for example, then, I loop in postfix that was calculated above to check every token and calculate them by using dictionary and push them back to another temporary list.



#### Picture 3.13: Calculating each token

And final step is that I tuple all sub-list in my list by the exact amount of operators then append them to list B so I can sum A and B together and give out result is a complete truth-table:



#### Picture 3.14: Final step

### 3.2.2 Manual running with test-cases:

* Testcase 1:
* Infix = ‘R|(P&Q)’

|  |  |  |
| --- | --- | --- |
| Current Symbol | Stack | Postfix String |
| R |  | R |
| | | | | R |
| ( | | ( | R |
| P | | ( | RP |
| & | | ( & | RP |
| Q | | ( & | RPQ |
| ) | | | RPQ& |
|  |  | RPQ&| |

* Return postfix string: ‘RPQ&|’
* A = [(False, False, False), (False, False, True), (False, True, False), (False, True, True), (True, False, False), (True, False, True), (True, True, False), (True, True, True)]
* B = [(False, False), (False, True), (False, False), (False, True), (False, False), (False, True), (True, True), (True, True)]
* Truthtable = [(False, False, False, False, False), (False, False, True, False, True), (False, True, False, False, False), (False, True, True, False, True), (True, False, False, False, False), (True, False, True, False, True), (True, True, False, True, True), (True, True, True, True, True)]
* Testcase 2:
* Infix = ‘~P|(Q&R)>R’

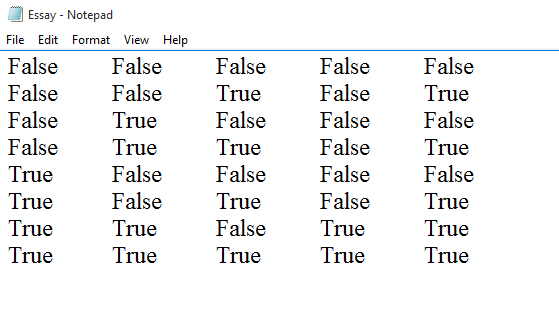
|  |  |  |
| --- | --- | --- |
| Symbol | Stack | Postfix |
| ~ | ~ |  |
| P | ~ | P |
| | | | | P~ |
| ( | | ( | P~ |
| Q | | ( | P~Q |
| & | | ( & | P~Q |
| R | | ( & | P~QR |
| ) | | | P~QR& |
| > | > | P~QR&| |
| R | > | P~QR&|R |
|  |  | P~QR&|R> |

* Return postfix string: ‘P~QR&|R>’
* A = [(False, False, False), (False, False, True), (False, True, False), (False, True, True), (True, False, False), (True, False, True), (True, True, False), (True, True, True)]
* B = [(True, False, True, False), (True, False, True, True), (True, False, True, False), (True, True, True, True), (False, False, False, True), (False, False, False, True), (False, False, False, True), (False, True, True, True)]
* Truthtable = [(False, False, False, True, False, True, False), (False, False, True, True, False, True, True), (False, True, False, True, False, True, False), (False, True, True, True, True, True, True), (True, False, False, False, False, False, True), (True, False, True, False, False, False, True), (True, True, False, False, False, False, True), (True, True, True, False, True, True, True)]

# CHAPTER 4: EXPERIMENTAL RESULTS

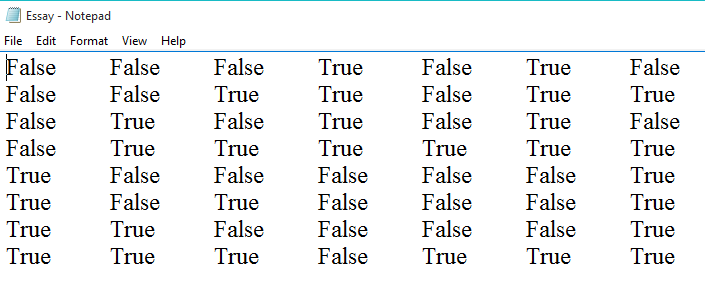
I usually use Python IDLE to compile my codes so below will be my screenshots of results after pressing F5 in Python IDLE or double click on Essay.py icon or right mouse then click ‘Open’:

* First test case: R|(P&Q)



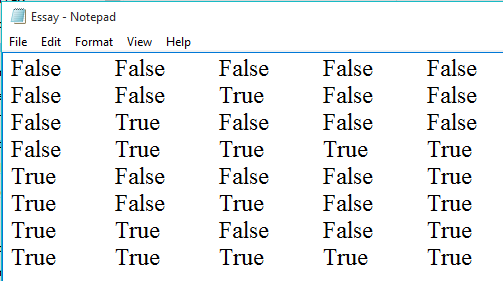
### *Picture 4.1: Screenshot test-case 1*

* Second test-case: ~P|(Q&R)>R



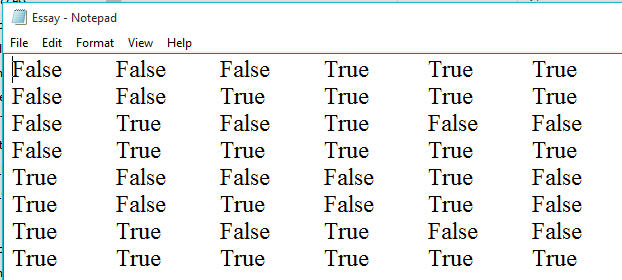
### *Picture 4.2: Screenshot test-case 2*

* Third test-case: P|(R&Q)



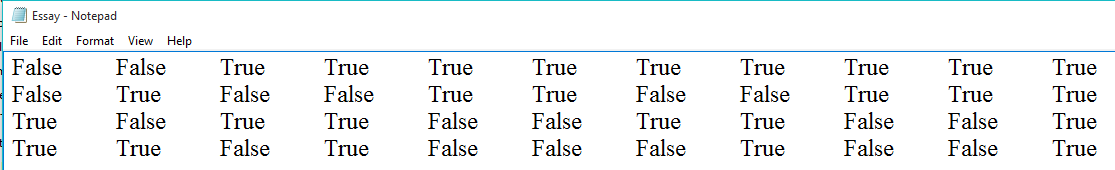
### *Picture 4.3: Screenshot test-case 3*

* Forth test-case: (P>Q)&(Q>R)



### *Picture 4.4: Screenshot test-case 4*

* Fifth test-case: (P|~Q)>~P=(P|(~Q))>~P



### *Picture 4.5: Screenshot test-case 5*

# CHAPTER 7: REFERENCES

1. *Social Network Analysis: Methods and Applications (Structural Analysis in the Social Sciences) – Stanley Wasserman, Katherine Faust*

<https://www.amazon.com/Social-Network-Analysis-Applications-Structural/dp/0521387078>

1. *Graph Theory with application – C.Vasudev*

<https://www.amazon.com/Graph-Theory-Applications-C-Vasudev/dp/812241737X>

1. *Social Networking Spaces\_ From Facebook to Twitter and Everything In Between A Step-by-Step Introduction to Social Networks for Beginners and Everyone Else – Todd Kelsey*

<https://www.amazon.com/Social-Networking-Spaces-Everything-Beginning/dp/B0085SL2F6>

1. *Medium.com*

<https://medium.com/@christina_meyer/what-is-the-difference-between-social-media-and-social-network-e6dd5ad28d8f>

1. *Investopedia.com*

<https://www.investopedia.com/terms/s/social-networking.asp>

1. *A writing about problems in social network analysis from a blurring – identity writer on* *skillicorn.wordpress.com:* <https://skillicorn.wordpress.com/2012/09/20/problems-at-the-heart-of-social-network-analysis/>
2. *Graph Theory and Social Networks Spring 2014 Notes by Kimball Martin*

<http://www2.math.ou.edu/~kmartin/graphs/graphs.pdf>

1. Applications of Social Media and Social Network Analysis - Przemysław Kazienko and Nitesh Chawla

<https://www.bookdepository.com/Applications-Social-Media-Social-Network-Analysis-Nitesh-Chawla/9783319364414>

1. *Sharing Emotion on Facebook: Network Size, Density, and Individual Motivation – Han Lin and Lin Qui*

<https://www.ntu.edu.sg/home/linqiu/publications/EmotionSharingOnFacebook.pdf>

1. Network Density definition in detail - <https://bookdown.org/omarlizardo/_main/2-9-density.html>