*“Heaven’s Light is Our Guide”*

Rajshahi University of Engineering & Technology, Rajshahi



Department of Electrical & Computer Engineering

(ECE-20)

Course Code: 2214

Course Title: Numerical Methods & Discrete Mathematics Sessional

**Lab Report 2: Converting English sentence into logical expression using python**

|  |  |
| --- | --- |
| Date of Experiment: 16-09-2023  Date of Submission: 25-09-2023 |  |
| *Submitted To*  **Md. Nahiduzzaman**  Lecturer  Dept. Of Electrical & Computer Engineering | *Submitted By*  Name: Md. Tajim An Noor  Roll: 2010025  Session: 2020-21 |

**Description:**

A **proposition** is a declarative sentence (that is, a sentence that declares a fact) that is either true or false, but not both. We can use letters and variables to represent a proposition. If a proposition is true or false, it can be represented by those letters or variables.

Logical sentences can be made be made with a single or multiple proposition. Depending on the sentence structure, they can be of many types, conjunctive, disjunctive, negation, conditional etc.

By converting a logical sentence into logical expression, we can make it easier for many people to understand some complicated sentences. They can appear hard to get, but by converting them to logical expression, we can find the final conclusion of a logical sentences, or we can at least find the possible answers for a complex propositional sentences.

In this experiment, the task is to find the logical expression with the letters and variables of a given sentence: **You can't ride the roller coaster if you are under 4 feet tall unless you are older than 16 years old.**

**Tools:**

-  MS Word

-  Python 3.10

- IDE (VS Code)

**Code:**

ans = "" #the final answer

negations = ["not", "unless", "n't"] #words that'll make a sentence negative

def check(a, x): #function to check if a sentence in positve or negative

for i in range(len(negations)):

if negations[i] in a:

return "¬" + x # adding negation sign

return x

def andOrLogic(a, b, k): #function to print the ans for and/or logical sentences

global ans

ans += check(a, "x")

if "and" in k:

ans += " v "

elif "or" in k:

ans += " ∧ "

ans += check(b, "y")

print(ans)

def ifLogic(k): # for logical sentences with if

global ans

if "if" and "unless" in k:

a, b = map(str, k.split("if"))

c, d = map(str, b.split("unless"))

ans += "(" + check(c, "r")

ans += " ∧ " + check(d, "s") + ")"

ans += " → " + check(a, "q")

elif "if" in k[0:2]:

if "then" in k:

a, b = map(str, k.split("then"))

elif "," in k:

a, b = map(str, k.split(","))

ans += check(a, "x")

ans += " → "

ans += check(b, "y")

elif "if" in k:

a, b = map(str, k.split("if"))

ans += check(b, "y")

ans += " → "

ans += check(a, "x")

print(ans)

def sentaceSplit(k): # funstion to split given sentence and splitting it to parts and find the logical expression

k = k.lower() #making all letters lower to avoid conflict

if "and" in k:

a, b = map(str, k.split("and"))

andOrLogic(a, b, k)

elif "or" in k:

a, b = map(str, k.split("or"))

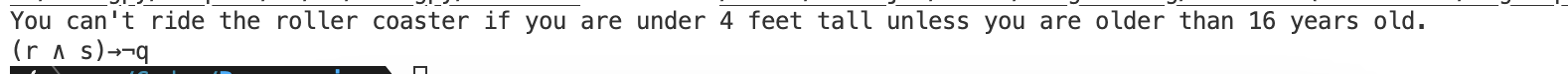
andOrLogic(a, b, k)

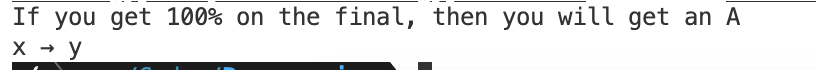
elif "if" in k:

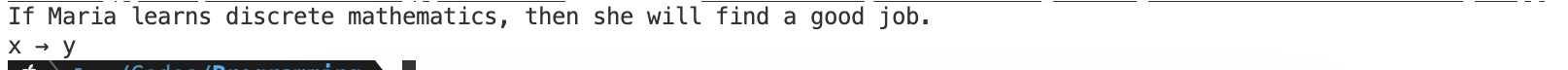
ifLogic(k)

sentenceSplit(input())

**Output:**





****

**Discussion:**

Here there’re 3 parts in the sentence. These three are propositions. To convert this into a logical expression, we first need to assign variables to each of the propositions. Then find if the propositions are individually false or true. It can be found using a function that will search for negative words in the propositions.

After that is done, the logical expression can be made.

For instance, for a simple two propositions separated by and, first split the sentence into two parts by using the map() function. Then pass both the parts in the function to find if they are positive or negative, then print the logical expression.

**Reference:**

* Discrete Mathematics and Its Applications 7th Edition by Kenneth H. Rosen