Computer Science Fundamentals

Seminar 4: Logic gates | Python: Lists, Loops

Lecturer: Olga Yugay

Seminar objectives

Part A

- Lecture based Q&A
- Logic gates exercises (Boolean expressions, logic diagrams, truth tables)

Part B

- PyCharm and GitHub
- Python
  - Lists
  - While loop
  - For loop

Part A: Logic gates

Lecture recap

Watch Boolean Logic & Logic Gates Crash Course Computer Science

The purpose of gates

With different combinations of logic gates, a computer performs

- the math (that is foundation of all computer operations)
- data storage

Lecture recap

1. Distinguish between a gate and a circuit
2. What are the three notational methods for describing the behavior of gates and circuits?
3. How many input signals can a gate receive and how many output signals can a gate produce?

For the following questions answer true or false

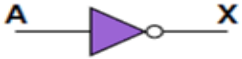
3. Boolean expressions are more powerful than logic diagrams in expressing the processing of gates and circuits.
4. Logic diagrams and truth tables are equally powerful in expressing the processing of gates and circuits.

Task 1

Name and explain the following gates. Provide corresponding boolean expressions:

a)

Logic Diagram Symbol

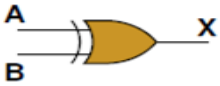


Truth Table

A	X
0	1
1	0

b)

Logic Diagram Symbol

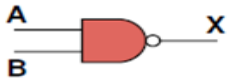


Truth Table

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

c)

Logic Diagram Symbol

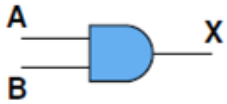


Truth Table

A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

d)

Logic Diagram Symbol

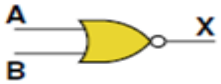


Truth Table

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

e)

Logic Diagram Symbol

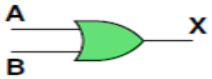


Truth Table

A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

f)

Logic Diagram Symbol



Truth Table

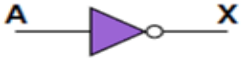
A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

Task 1: answers

Name and explain the following gates. Provide corresponding boolean expressions:

a)

Logic Diagram Symbol



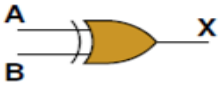
x=A’

Truth Table

A	X
0	1
1	0

b)

Logic Diagram Symbol



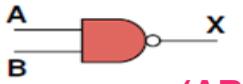
x=A⊕ B

Truth Table

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

c)

Logic Diagram Symbol



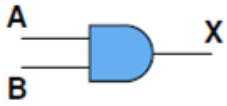
x=(AB)’

Truth Table

A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

d)

Logic Diagram Symbol



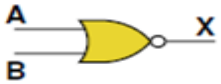
x=AB

Truth Table

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

e)

Logic Diagram Symbol



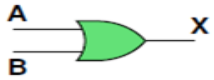
x=(A+B)’

Truth Table

A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

f)

Logic Diagram Symbol



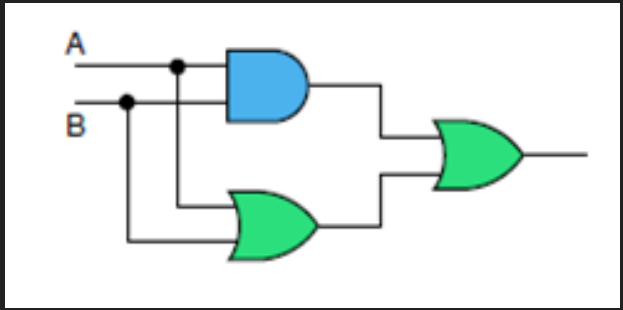
x=A+B

Truth Table

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

Task 3

Write a boolean algebra expression and show the behavior of the following circuit with a truth table:

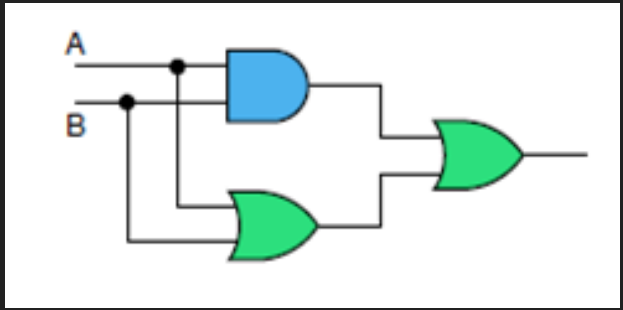


You may then check the answer using the simulator

https://academo.org/demos/logic-gate-simulator/

Task 3 answers

Write a boolean algebra expression and show the behavior of the following circuit with a truth table:



A	B	AB	A+B	AB + (A + B)
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	1

You may then check the answer using the simulator

https://academo.org/demos/logic-gate-simulator/

Boolean algebra simplification

- Boolean Algebra Logic Circuit Simplification

[1] Khan Academy and Code.org, 2018, Circuits & Logic. [online] Available at: <https://youtu.be/Sc3lh3D4rCw >

Part B: Python

Topics to cover

- Lists
- For loop
- While loop

Lists

list is a sequence of values. In a string, the values are characters; in a list, they can be any type. The values in a list are called elements or sometimes items.

[10, 20, 30, 40]

['crunchy frog', 'ram bladder', 'lark vomit']

['spam', 2.0, 5, [10, 20]]

List examples

```
>>> cheeses = ['Cheddar', 'Edam', 'Gouda']

>>> numbers = [42, 123]

>>> empty = []

>>> print(cheeses, numbers, empty)

['Cheddar', 'Edam', 'Gouda'] [42, 123] []
```

List examples

The expression inside the brackets specifies the index. Remember that the indices start at 0

```
planets = [‘mercury’, ‘venus’, ‘earth']
```

```
>>> planets[0]
‘mercury'
```

Unlike strings, lists are mutable. When the bracket operator appears on the left side of an assignment, it identifies the element of the list that will be assigned.

```
>>> numbers = [42, 123]
```

```
>>> numbers[1] = 5
```

```
>>> numbers [42, 5]
```

List examples

The in operator also works on lists.

```
>>> cheeses = ['Cheddar', 'Edam', 'Gouda']

>>> 'Edam' in cheeses

True

>>> 'Brie' in cheeses

False
```

Task 5

Write a Python program to display the **first** and **last** colors from the following list.

color_list = ["Red","Green","White" ,"Black"]

Commit with message task 5 and push to Github

Iteration

Iteration is the ability to run a block of statements repeatedly.

- While loop
- For loop

while loop

With the while loop we can execute a set of statements as long as a condition is true.

```
i = 1

while i <=10:

    print (i)

    i += 1
```

Task 6

Update the previous while loop to display

```
*
**
***
****
*****
******
*******
********
*********
**********
```

Commit with message task 5 and push to Github

Task 7

Update the code to display

```
**********
*********
********
*******
******
*****
****
***
**
*
```

Commit with message task 6 and push to Github

while

With the while loop we can execute a set of statements as long as a condition is true.

```
def countdown(n):
    while n > 0:
        print(n)
        n = n - 1
    print('Blastoff!')
```

break

Sometimes you don’t know it’s time to end a loop until you get halfway through the body. In that case you can use the break statement to jump out of the loop.

For example, suppose you want to take input from the user until they type done. You could write:

```
while True:
    line = input('> ')
    if line == 'done':
        break
    print(line)
print('Done!')
```

Task 8

Update previous code to guess the secret number

Commit with message “ task 7 Guess secret number” and push to Github

For loop

A for loop is used for iterating over a sequence (that is either a list, a tuple, a dictionary, a set, or a string).

```
#Example 1
subjects= ["CSF", "FunPro", "WT"]
for x in subjects:
  print(x)

#Example 2
for x in range(6):
  print(x)
```

range() is a function. More about it here

range function

range(start, stop, step)

Parameter	Description
start	Optional. An integer number specifying at which position to start. Default is 0
stop	Required. An integer number specifying at which position to stop (not included).
step	Optional. An integer number specifying the incrementation. Default is 1

```
#Example 1
x = range(6)
for n in x:
  print(n)
```

```
#Example 2
x = range(3, 6)
for n in x:
  print(n)
```

```
#Example 3
x = range(3, 20, 2)
for n in x:
  print(n)
```

Task 9

Write a Python program to construct the following pattern, using a nested for loop.

```
*
* *
* * *
* * * *
* * * * *
* * * *
* * *
* *
*
```

Part of the code is given below:

```
n=10;
for i in range(n):
    for j in range(i):
        print ('* ', end="")
    print('')
```

The parameter to end line, by default it is \n

Commit with message task 8 and push to Github

Review Run-length encoding given last seminar

```
def encode(message):
    encoded_string = ""
    i = 0
    while (i <= len(message)-1):
        '''first while loop is for going through each letter'''
        count = 1
        ch = message[i]
        j = i #assign current letter index to j in order to count the occurrence of letters
        print(ch)
        print('this is j', j)
        print('this is i ', i)
        while (j < len(message)-1): # to loop from particular letter till the end
'''if char at the current index is the same as char at next index, the count is incremented by 1'''
            if (message[j] == message[j + 1]): #if letters are equal to each other, keep looping, else break
                count = count + 1
                j = j + 1
            else:
                break
        '''the count and the character is concatenated to the encoded string'''
        encoded_string = encoded_string + str(count) + ch
        i = j + 1
    return encoded_string
encoded_message = encode("ABBBBCCCCCCCCAB")
print(encoded_message)
```

Think of **decode** function logic and push to /TW5

Challenge task:

Think of **run length decode function** logic and write the code.

Commit with message “Run-length decode task” and push to Github

Once done compare solution with Aluda, T., 2021. Run Length Encoding (RLE) Compression Algorithm in Python. [online] Engineering Education (EngEd) Program | Section. Available at: https://www.section.io/engineering-education/run-length-encoding-algorithm-in-python

Homework

Write a Python program to print alphabet pattern 'D'.

Expected Output:

```
****
*  *
*  *
*  *
*  *
*  *
****
```

Push code to your repository (for example seminar5/) on Github.

Homework

- Dale, Computer Science Illuminated, **Chapter 4, end of the book exercises**
- Watch
  - Boolean Logic & Logic Gates Crash Course Computer Science #3
  - How Computers Calculate - the ALU: Crash Course Computer Science
  - Registers and RAM Crash Course Computer Science
- Read Ron White, How Computers Work, pp 34-35 and try to understand how Processor does the math

- Explore how truth tables can be used to analyze a problem, Computer Science Distilled, Filho pp.10-12

- Khan Academy. (2011, June 30). For loops in Python. [Video]. YouTube.

- Khan Academy. (2011, June 30). While loops in Python. [Video]. YouTube.

- Downey, Think Python, Chapter 10, end of the Chapter exercises

- https://www.jetbrains.com/help/pycharm/using-git-integration.html

References and recommended sources

1. Dale, N., & Lewis, J. (2020). Computer Science Illuminated (7th ed.). Jones & Bartlett Learning, Chapter 4
2. Downey, A. (2015). Think python. O'Reilly. Chapter 1, 2
3. Hamedani, M., 2020. Python for Beginners - Learn Python in 1 Hour. [online] Youtube.com. Available at: <https://www.youtube.com/watch?v=kqtD5dpn9C8> [Accessed 5 June 2022].
4. Aluda, T., 2021. Run Length Encoding (RLE) Compression Algorithm in Python. [online] Engineering Education (EngEd) Program | Section. Available at: <https://www.section.io/engineering-education/run-length-encoding-algorithm-in-python/> [Accessed 6 June 2022].