```python
result = 10 // 3
print(result)     # Output: 3
```

9.2. Modulo (`%`)
Syntax: `dividend % divisor`
```python
result = 10 % 3
print(result)     # Output: 1
```

10. Generating a List with `list(range())`
Syntax: `list(range(start, stop, step))`
```python
# Generate a list of numbers from 0 to 4
my_list = list(range(5))
print(my_list)    # Output: [0, 1, 2, 3, 4]
```

12. Logical Operators
- Logical Function: XOR
  - Python Operator: `^`
  - Description: Returns True if exactly one operand
is True (bitwise XOR)
- Logical Function: NAND
  - Python Operator: `not (a and b)`
  - Description: Returns False only if both operands
are True
- Logical Function: NOR
  - Python Operator: `not (a or b)`
  - Description: Returns True only if both operands
are False
- Logical Function: XNOR
  - Python Operator: `not (a ^ b)`
  - Description: Returns True if both operands have
the same boolean value
- Logical Function: Implication
  - Python Operator: `not a or b`
  - Description: Returns False only if the first
operand is True and the second is False

```python
a = True
b = False

print(a and b)   # False
print(a or b)    # True
print(not a)     # False
print(a ^ b)     # True
print(not (a and b))  # True (NAND)
print(not (a or b))   # False (NOR)
print(not (a ^ b))    # False (XNOR)
print(not a or b)     # False (Implication)
```

True Table
```python
values = [True, False]
print("Truth Table for XOR:")
print("{}\t{}\t{}".format("a", "b", "a xor b"))
for a in values:
    for b in values:
        print("{}\t{}\t{}".format(a, b, xor(a, b)))
```

13. `pare_line`
```pyhon
>>> print(parse_line("Meka's Lounge |
Bars|5|2|4|4|3|4|5"))
[7, 3.857142857142857, "Meka's Lounge", "Bars"]
```

```python
def parse_line(string):
    parsed = string.split(',')
    average = 0
    total = len(parsed)-2
    for i in range(2,len(parsed)):
        average += int(parsed[i].strip())
    average /= total
    return [total, average, parsed[0].strip(),
parsed[1].strip()]
```

13.1 filter restaurant
```python
infile = input("Input Filename? ")
outfile = input("Output Filename? ")
lines = []
file = open(infile, "w")
for line in open(infile):
    parsed = parse_line(line)
    if parsed[0] > 3 and parsed[1] > 2.5:
        file.write(parsed[2]+'\n')
file.close()
```

Chess Board
```python
def move_piece(board, piece, start, end):
    if board[start[0]][start[1]] == piece:
        board[start[0]][start[1]] = ''
    taken = board[end[0]][end[1]]
        board[end[0]][end[1]] = piece
        return taken
    return
```