Required Software

- Python 3 (Latest Version)
- Jupyter Notebook
- Anaconda

Python I/O

- We use the print() function to output data to the standard output device (screen).
  - print('Hello World!')

- The input() method reads a line from input, converts into a string, and returns it.
  - input('Enter anything ')

Python Variables

- Variables are like a container for storing data.
- Compares to other programming languages, Python has no command for declaring a variable.
- A variable is created the moment you first assign a value to it.

Example:

Var = ‘data science’

Var2 = ‘study mart’

List of Keywords in Python: https://www.programiz.com/python-programming/keyword-list

Python Variables

A variable can have a short name (like x and y) or a more descriptive name.

- Keywords can't use as a variable.
- A variable name must start with a letter or the underscore (_) character.
- A variable name cannot start with a number.
- A variable name can only contain alpha-numeric characters and underscores (A-Z, 0-9, and _).
- Variable names are case-sensitive (x, X, _x are three different variable).

Valid Example:

```
Var = 10
Var2 = 100
_var = 20
Var_2 = 10
V1a2r3 = 30
My_name = 'shakil'
```

Invalid Example:

```
9Var = 'data science'
Var-2 = 'study mart'
&var = 20
My name = 'shakil'
```

Python Variables

Multiple

- Multiple Variables:
  - x, y, z = “Data", “Science", “Smart“ -> Valid
  - x, y, z = “Data", “Science“ -> Invalid
- Comments:
  - Single Line
  - Multiple Line

Python Variables

Local Vs. Global

- Multi Word Variable Name
  - camelCaseVar
  - PascalCaseVar
  - snake_case_var

- Global Variable: Variables that are created outside of a function are known as global variables. Global variables can be used by everyone, both inside of functions and outside.

- Local Variable: Variables that are created inside of a function are known as local variables. local variables can be used by inside of function.

All about Python Strings

X = ‘Data Science’

Y = ’10’

Z = Something

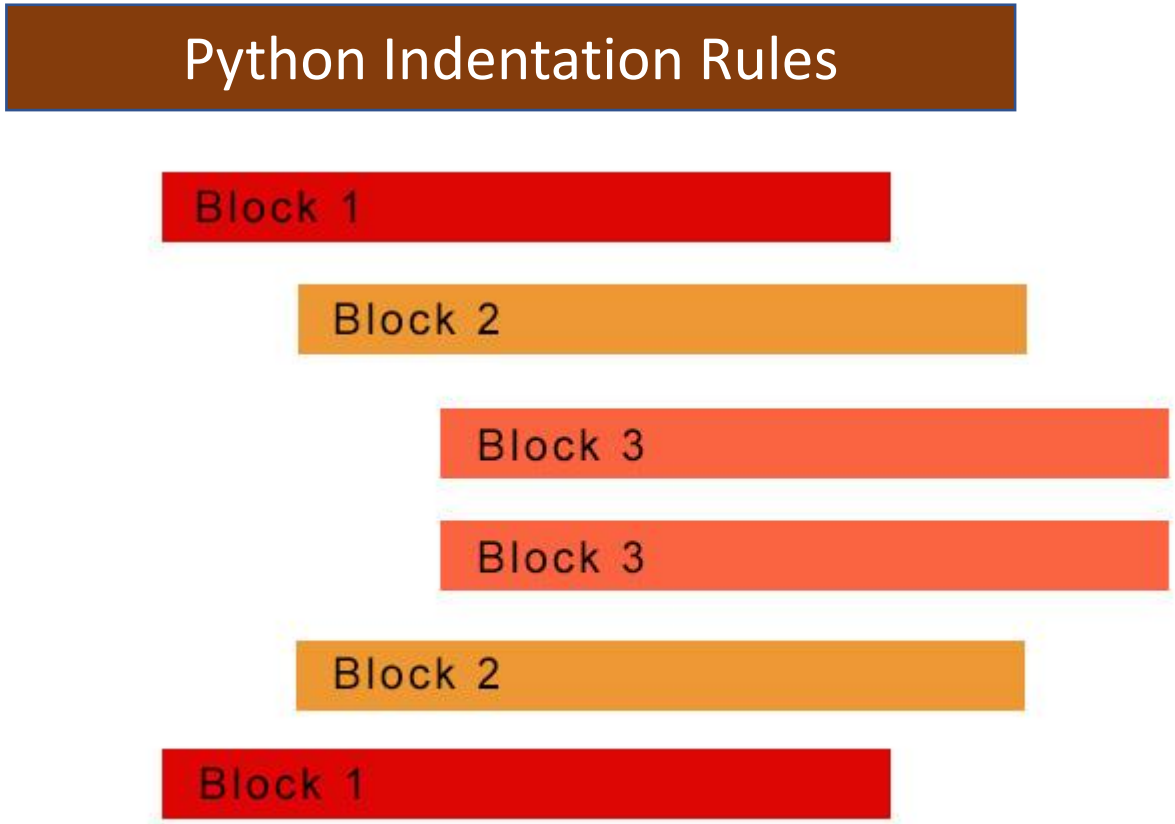
- String Formatting
- String Concatenation
- String methods

Python supports the usual logical conditions from mathematics:

- Equals: a == b
- Not Equals: a != b
- Greater than a > b
- Greater than or equal to a >= b
- Less than a < b
- Less than or equal to a <= b

Conditional Statements

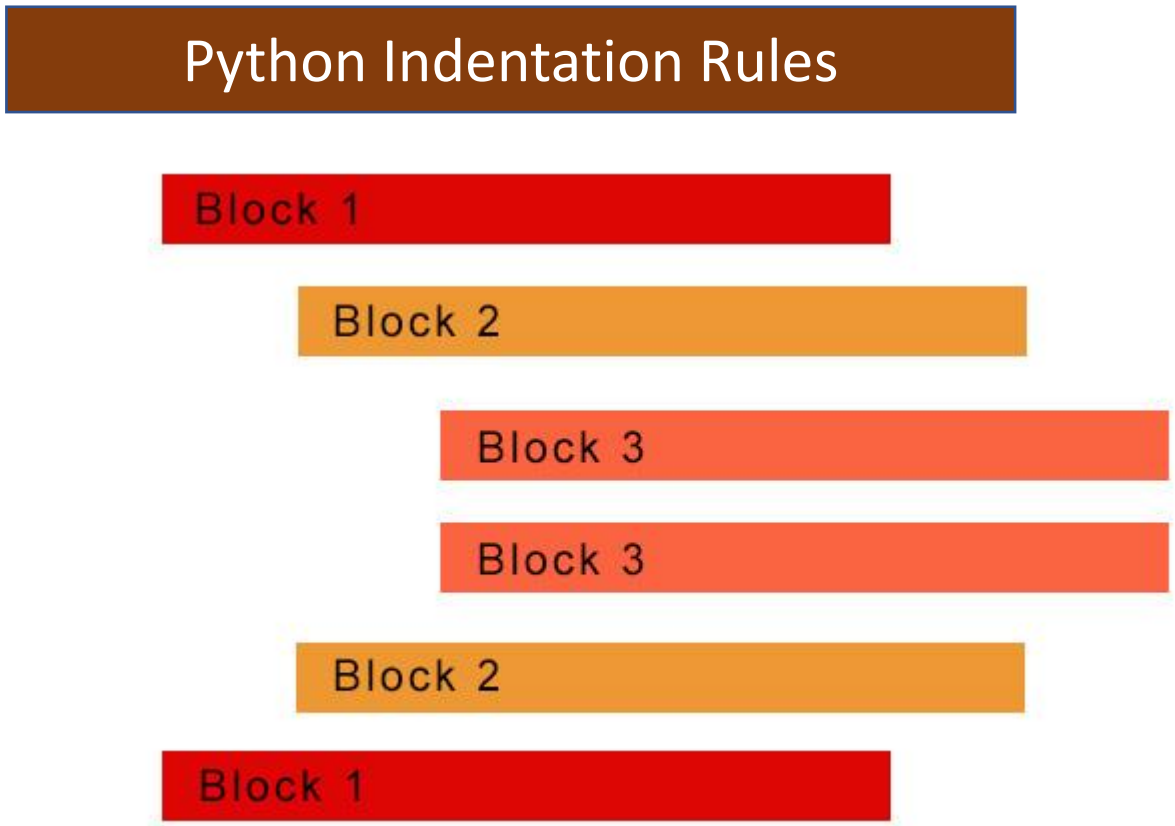
If, else




```
x = 50
y = 100

if y > x:
    print("y is greater than x")

elif x == y:
    print(" x and y are equal")

else:
     print(" x is y greater than y ")
```

Conditional Statements

If, else




```
scores = [85, 92, 78, 60, 45]

for score in scores:
    if score >= 90:
        grade = "A"
    else:
        if score >= 80:
            grade = "B"
        else:
            if score >= 70:
                grade = "C"
            else:
                if score >= 60:
                    grade = "D"
                else:
                    grade = "F“

    print(f"Score: {score}, Grade: {grade}")
```

Example

```
subjects = [“ai", “data science", “statistics“, ”math”]
for x in  subjects :
            print(x)
```

Output:

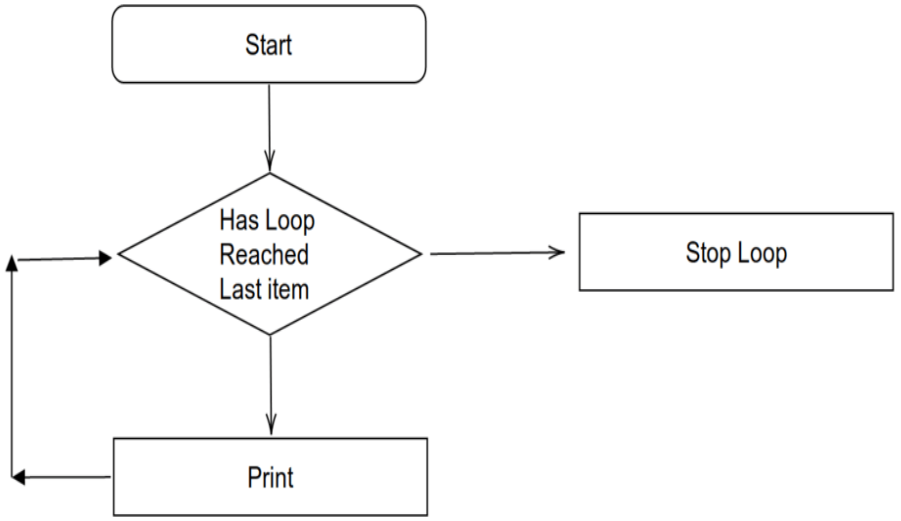
“ai“

“data science“

“statistics“

”math”

For Loop



Loops

while

```
counter = 1

while counter <= 5:
    print(counter)
    counter += 1
```