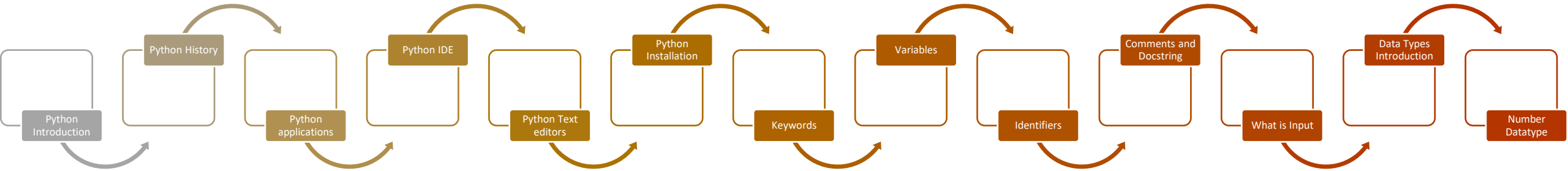
Python for Data Science

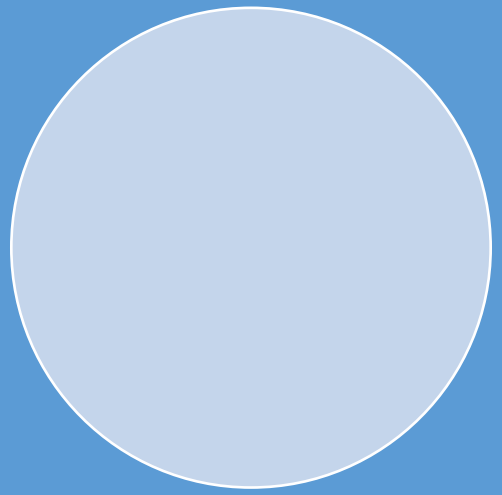
Python Divides 3 Different Phases


Python Basics Series
Python Intermediate Series
Python Advanced Series

Module -1
Python Basics Series

Course of Content





PYTHON INSPIRED FROM

ABC LANGUAGE

C LANGUAGE

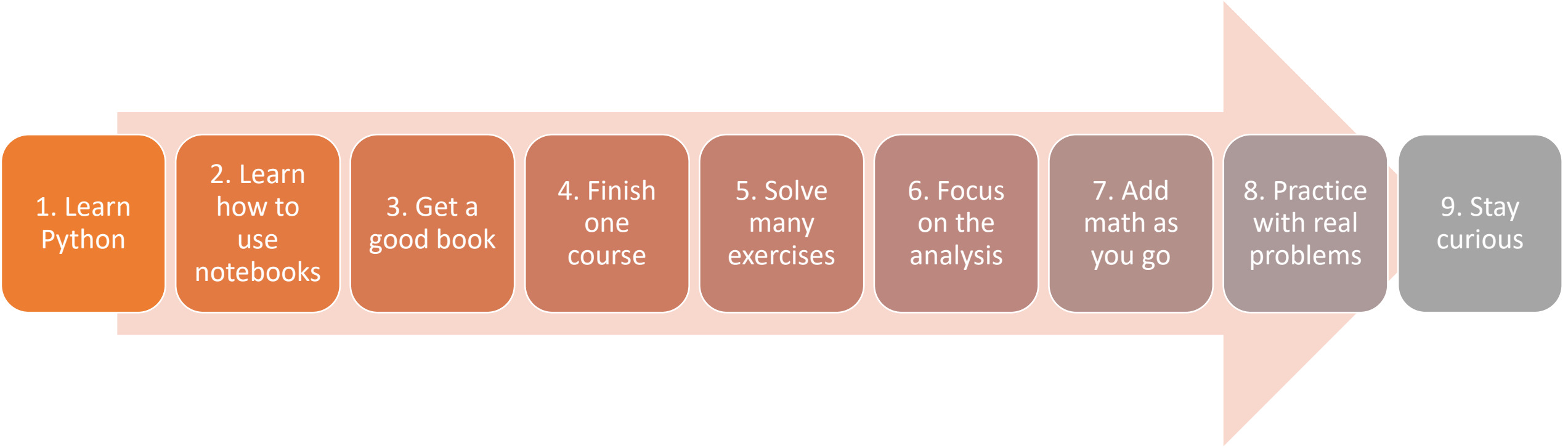
MODULA-2+

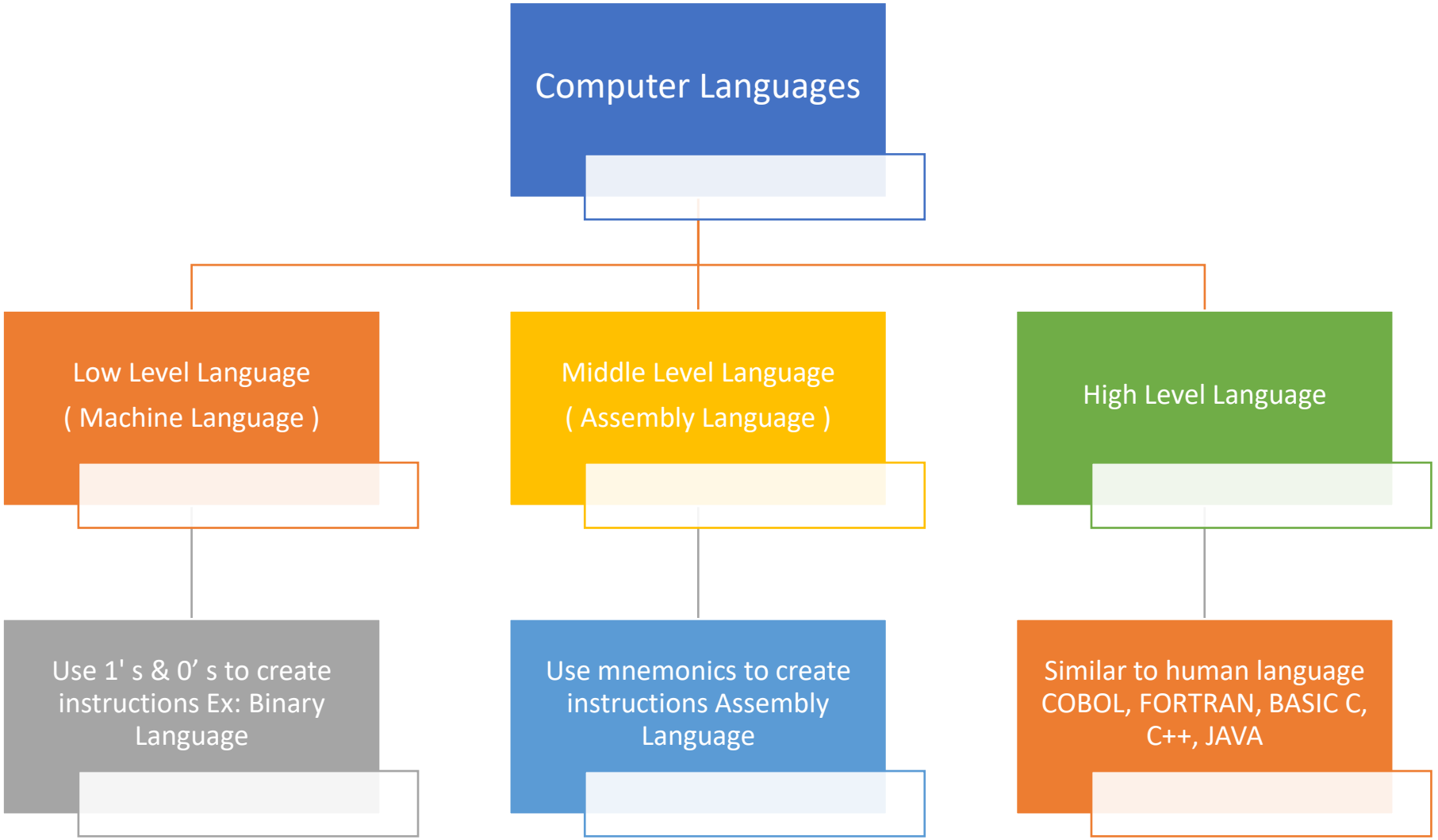
ALGOL 68

ICON

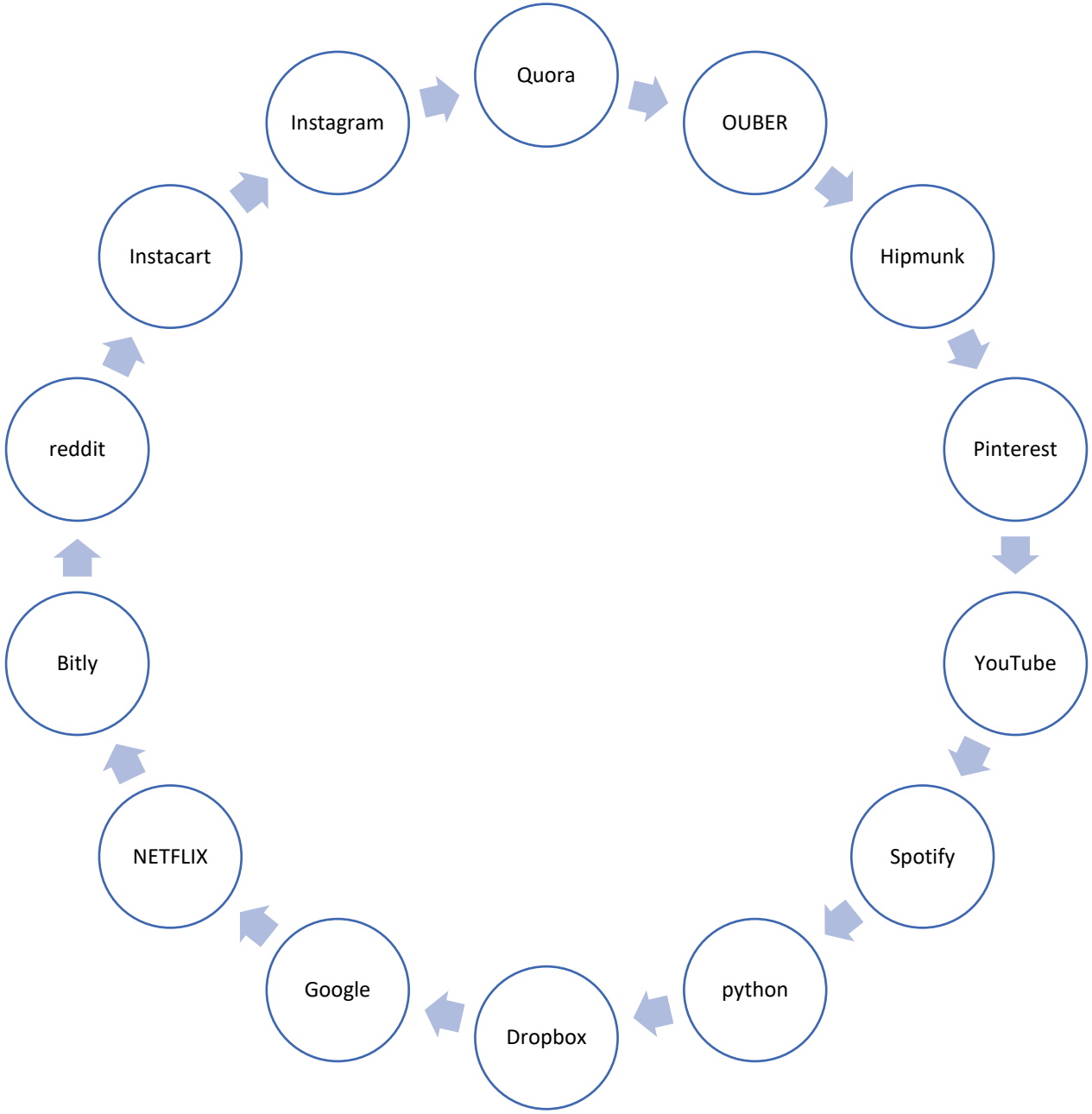
BOURNE SHELL

Are you ready to dive into the exciting world of data science and machine learning? Follow this comprehensive roadmap to success:



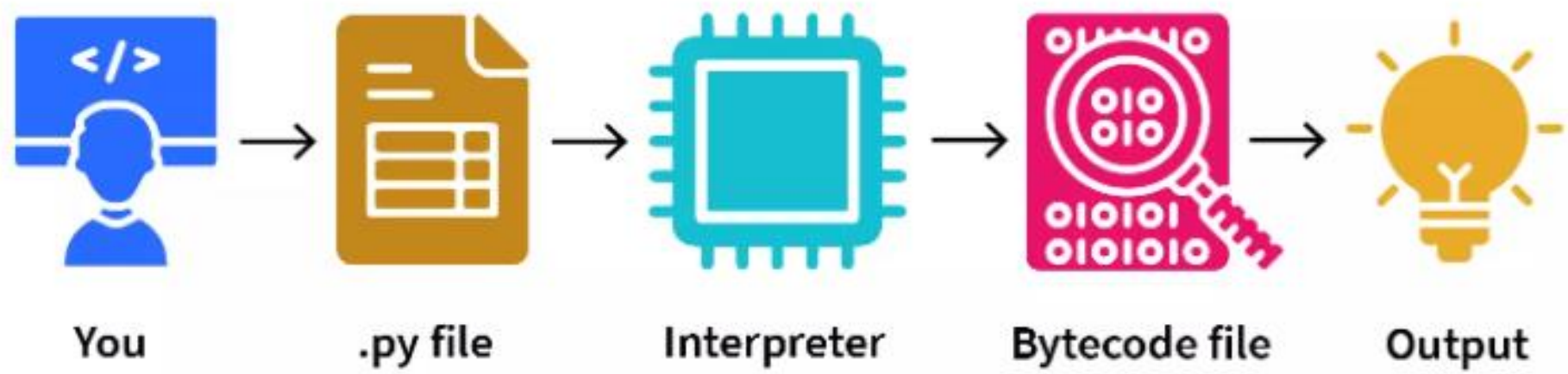

Computer Languages
Low Level Language
(Machine Language)
Middle Level Language
(Assembly Language)
High Level Language
Use 1' s & 0’ s to create instructions Ex: Binary Language
Use mnemonics to create instructions Assembly Language
Similar to human language COBOL, FORTRAN, BASIC C, C++, JAVA

Who Uses Python



Why Python

What is PYTHON ?



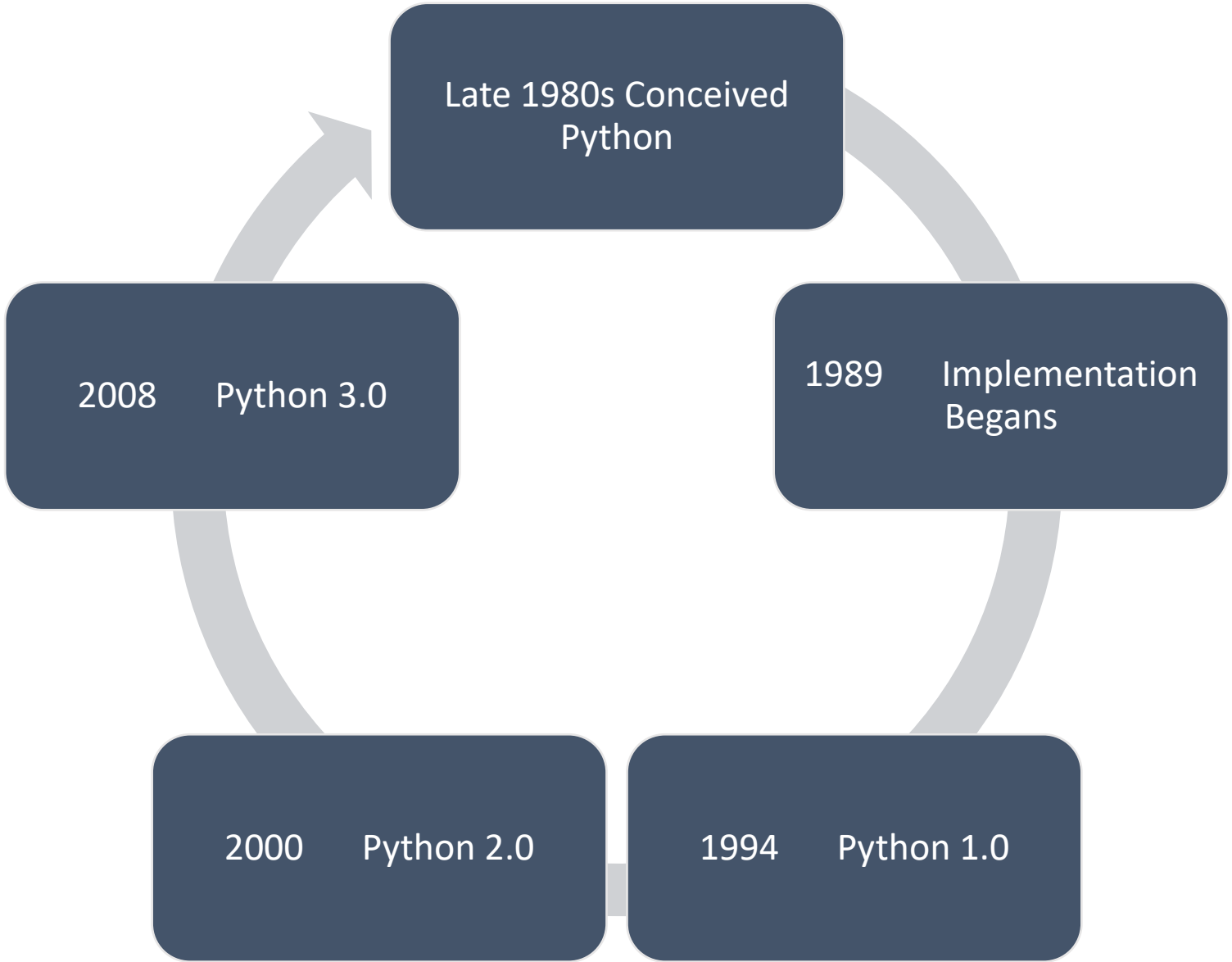
Introduction to Python

Development of Python

>Python was developed in the late 1980s and its implementation was started in December 1989 by Guido Van Rossum at CWI (Centrum Wiskunde & Informatica) in the Netherlands.

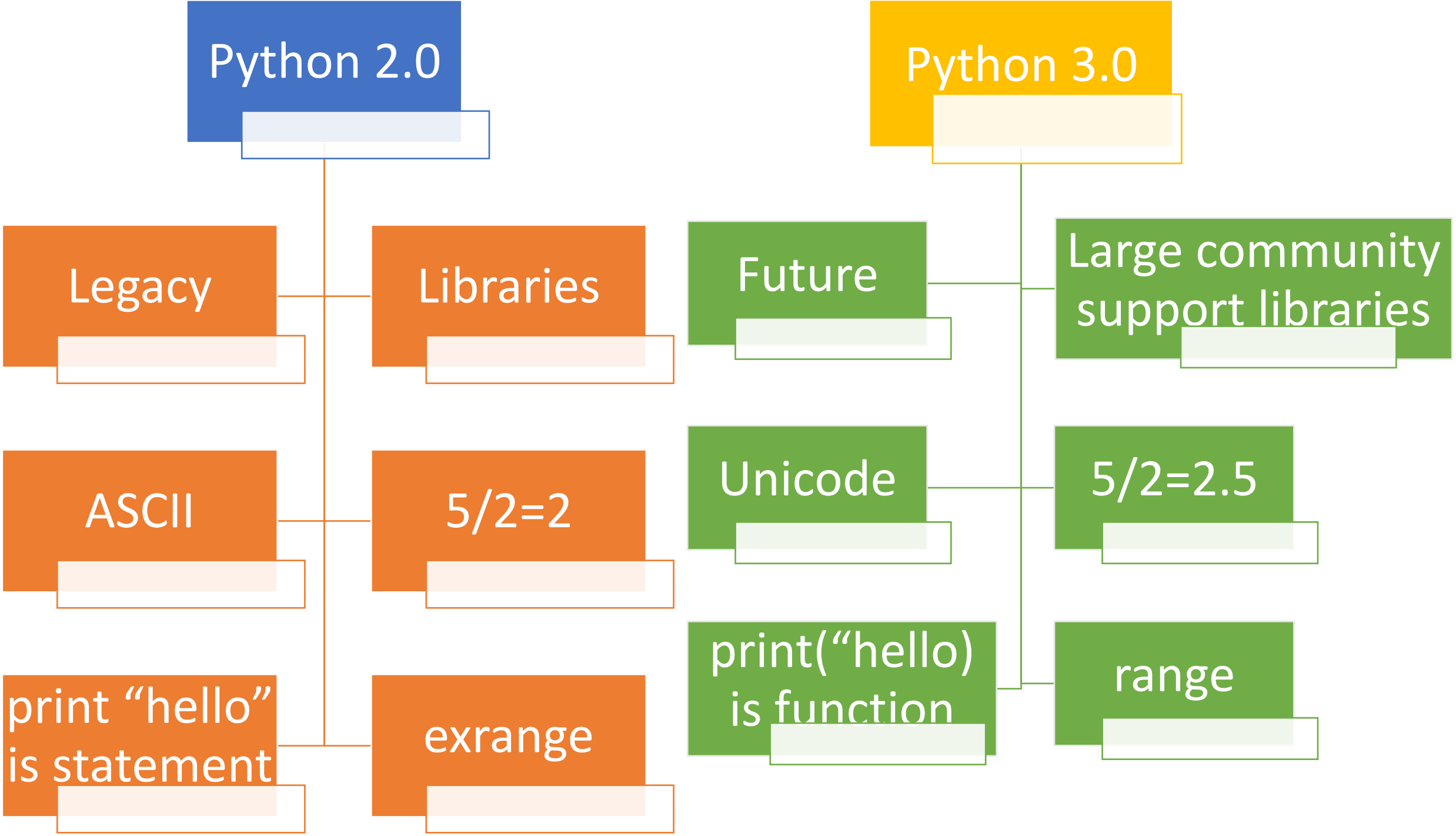
Created in 1991 by Guido Van Rossum

Python Programming Language Brief History

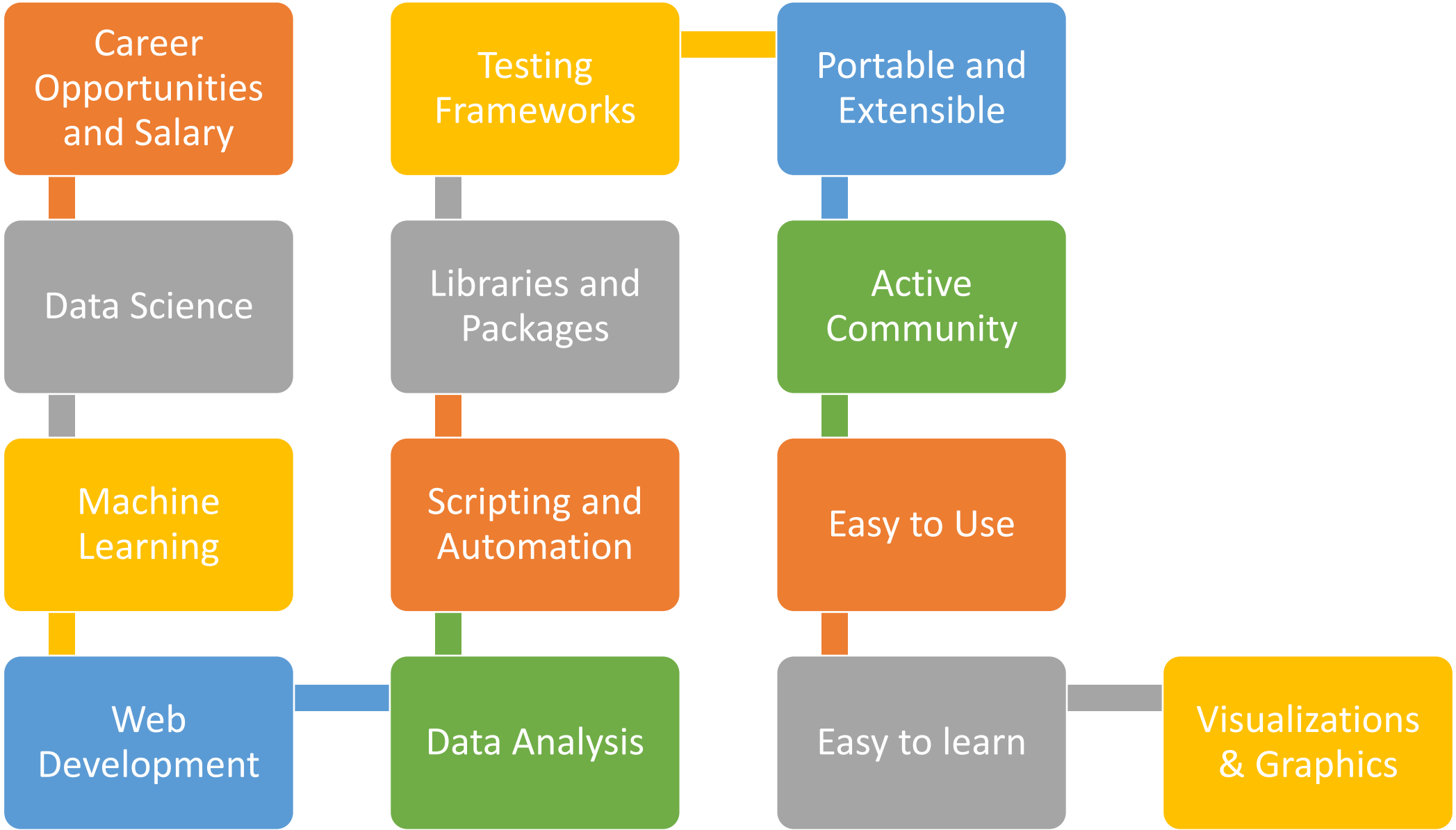


Features of Python

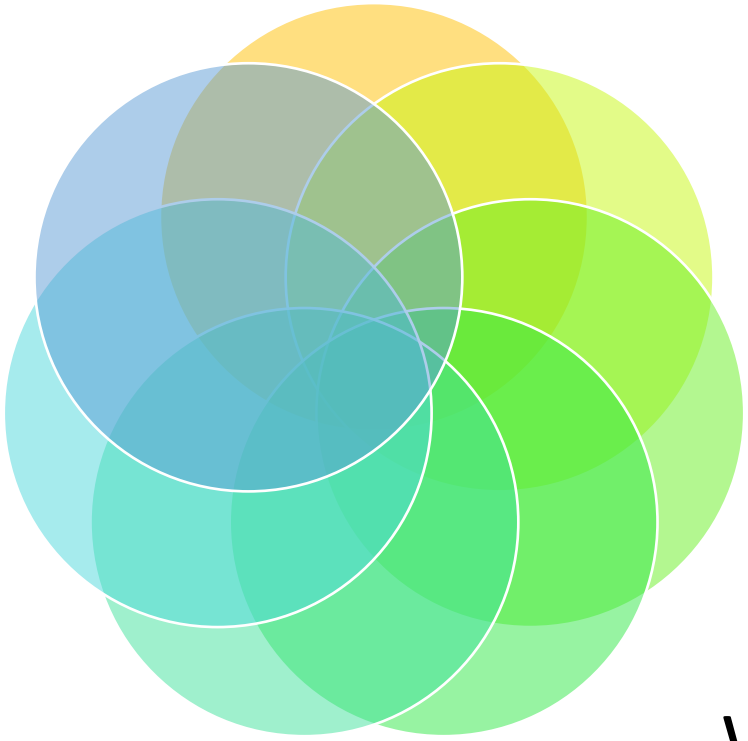
- Binary files to Table
- Python is not domain specific language
- Python is general purpose language
- High Level Programming Language
- python is interpreted language
- Python is Object Oriented Programming Language
- Classes and Objects
- Code reusability
- Beginner friendly.
- Open source.
- Third Party Libraries.
- Large Community version.


Python 2.0
Legacy
Libraries
ASCII
5/2=2
print “hello”
is statement
exrange
Python 3.0
Future
Large community
support libraries
Unicode
5/2=2.5
print(“hello)
is function
range

Why should you learn Python?

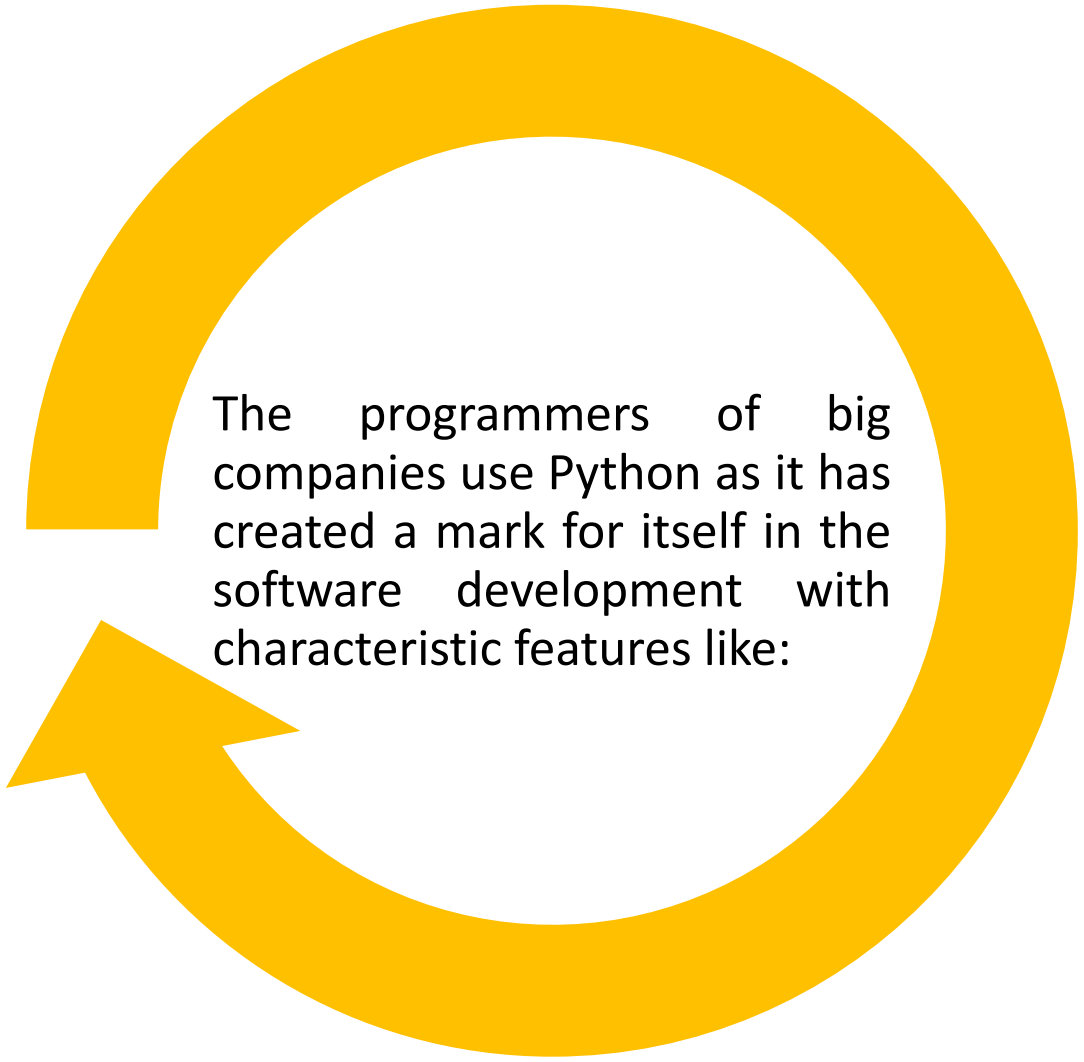


Cons of python



Popularity of Python

Why Companies Prefer Python?



- Interactive
- Interpreted
- Modular
- Dynamic
- Object-oriented
- Portable
- High level
- Extensible in C++ & C

Python is compiler or interpreter?

The Answer is ***both***

Java vs C vs Python

Java

```
class welcome {
public static void main(String [ ] args) {
System.out.println (“ Hello World ”) ;
}
}
```

C

```
#include <stdio.h>

int main() {
   printf("Hello World!");
   return 0;
}
```

Python

```
print (‘ hello world‘ )
```

Working In Python

Python is free open-source software that works on Linux Mac Windows and various other platforms (21 in total).

- It comes preinstalled on Mac and most distributions of Linux.
- There are multiple python distributions available.

Flavours of Python

PyPy: Implemented In python.

CPython: Written in C the most common implementation of Python.

Jython: Written in lava compiles to bytecode.

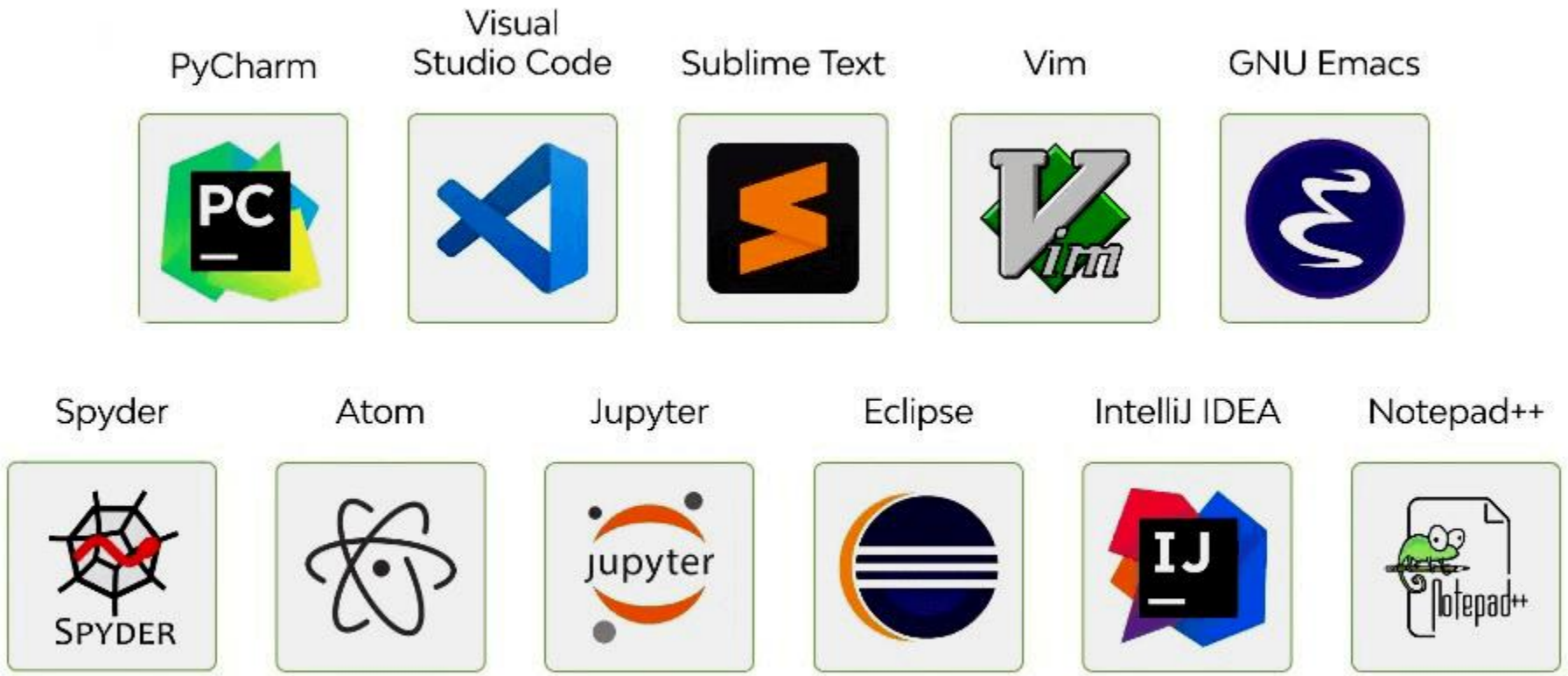
MicroPython: Runs on a microcontroller.

IronPython: Implemented in c# an extensibility layer to frameworks written in .NET.

Brython: Browser Python runs in the browser.

RubyPython: Bodge between Python and Ruby interpreters.

Best Python IDE's and Editors



Other IDEs

- Ninja IDE
- Django
- Pyscripetr
- Beeware
- Coders diaries
- Eric
- Exedore
- Wing
- Pydev
- Pyside
- Pyzo
- Rubymine
- Rodeo

Other Editors

- Komodo
- Emacs
- Gedit
- BBEDIT
- Bluefish
- Editra
- Lio
- Textmate
- Textwrangler

What are the basic modes of python?

Python has two basic modes:

1. Script mode.
2. Interactive mode.

Python Installation

->Python Set Path in Windows
->goto environment set variable

1. Program
2. Script

Documentation commands

help()
help>topics
help>LISTS
helps> quit
help('LISTS')

How To Run Programs?

I. Script & Interpreter.

II. Different Ways To Run Python Script Through .

a. Microsoft Command Prompt
b. The Python Interactive Mode
c. Using The Python Command Line
d. Run On Text Editor
e. Run On IDLE
f. Run On IDE (PyCharm)

Modules Packages Libraries & Framework

Agenda

The smallest individual unit in a program is known as tokens.

Python has following tokens:-

1. Keywords
2. Identifiers
3. Literals
4. Operators
5. Punctuators
+-%/ =

Python is static or dynamic?

The Answer is ***Dynamic.***

Keywords in Python

Keywords are the reserved words in Python. We cannot use a keyword as a variable name function name or any other identifier.

Identifier in Python

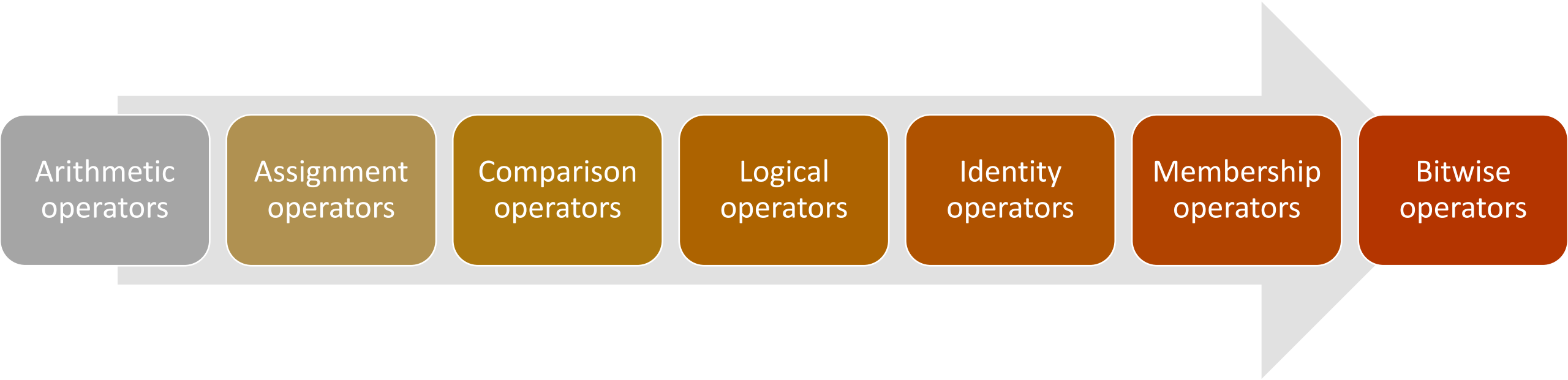
Python Identifiers are user-defined names They are used to specify the names of variables functions class module and objects.

Literals in Python

Literals in Python are nothing but a succinct way of representing the data types In simpler words it is the way to represent a fixed value in our source code. They can either be numbers text Boolean or any other form of data.

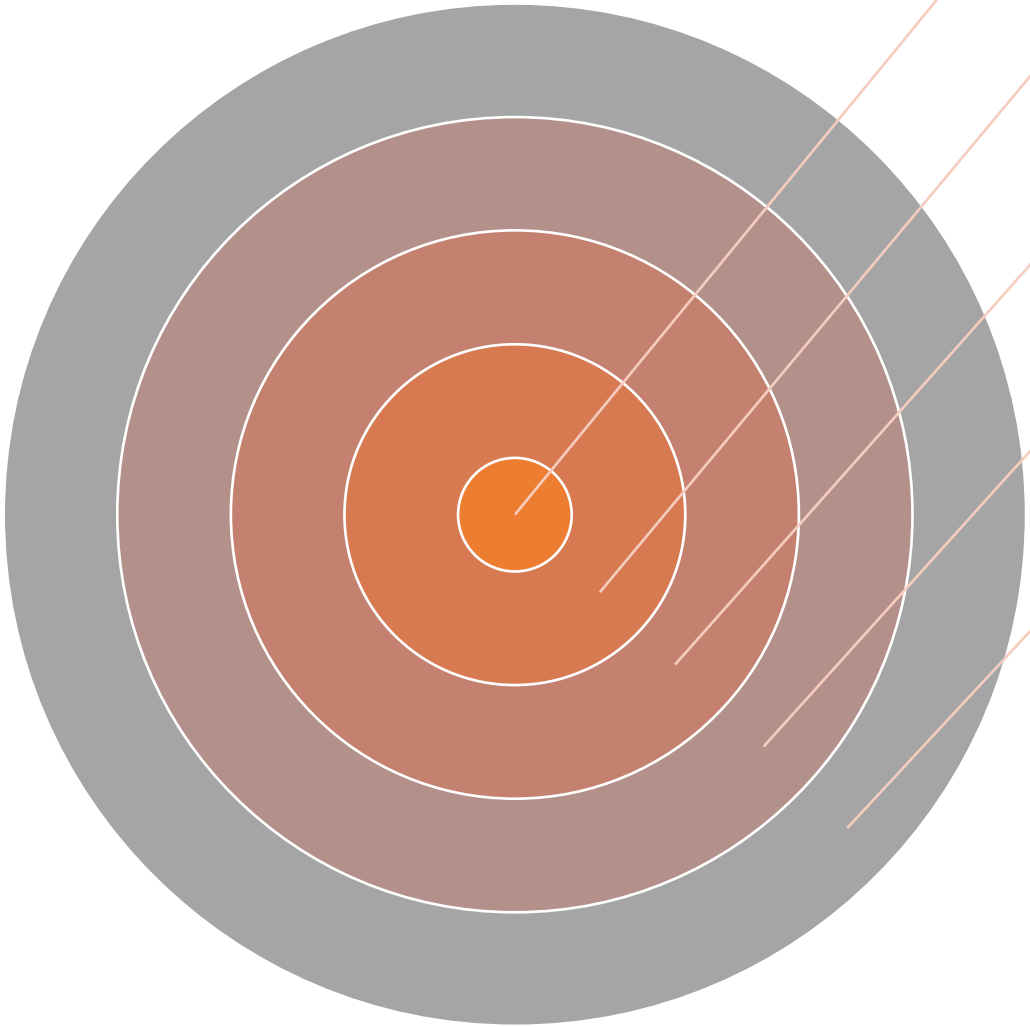
Operators in Python:

Operators in Python are special symbols that carry arithmetic or logical operations. The value that the operator operates on is called the operand.



Punctuators in Python:

These are the symbols that used in Python to organize the structures statements and expressions. Some of the Punctuators are: [ ] { } ( ) @ -= += *= //= **== = etc.



What are **variables**?

Why we use **variables**?

Use of **variables**?

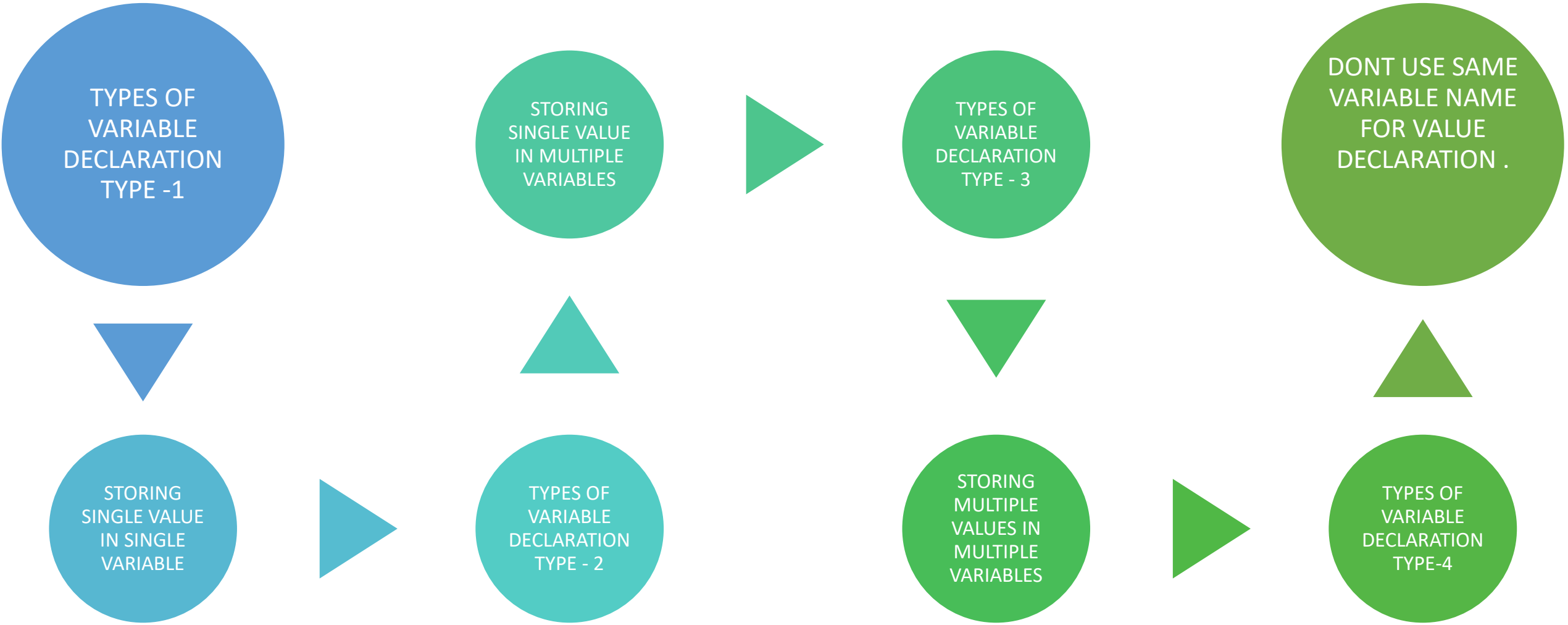
How to declare **variable**?

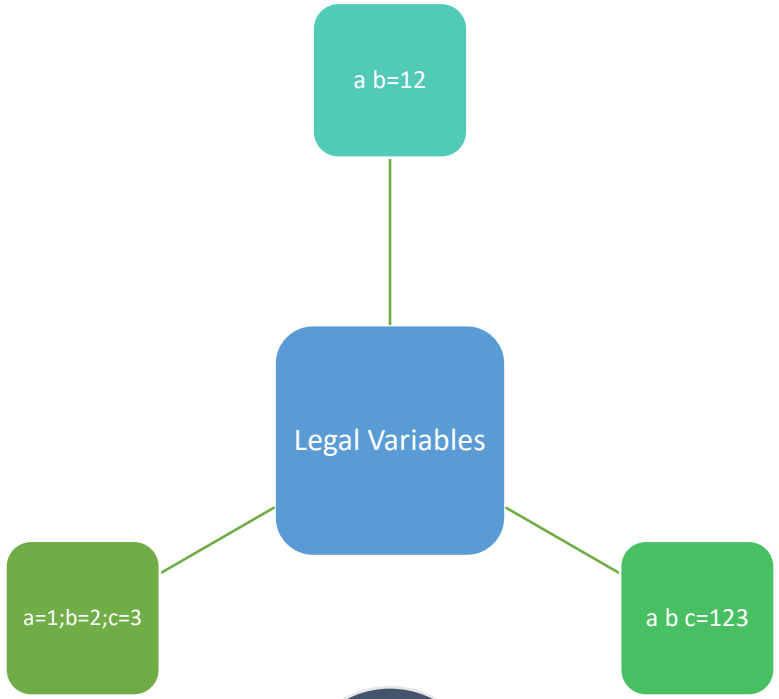
How many ways we declare **variable**?

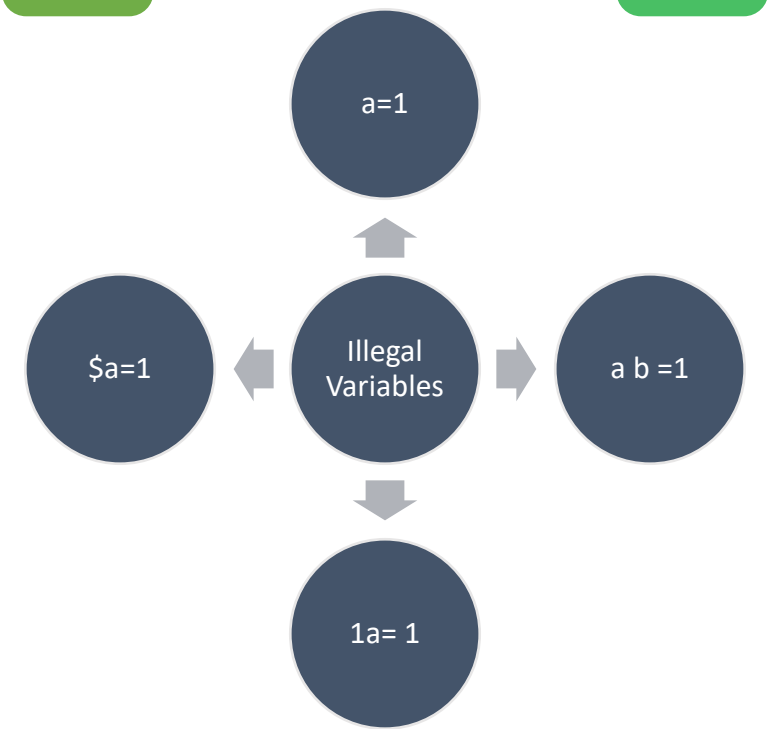
Rules For Creating Variables :

1. Must Starts With Letters (Upper Or Lower Case) Underscore.
2. Variable name Cannot Start With Numbers.
3. Variable name Contains Letters Underscore Numbers.
4. Special Characters Are Not Allowed Except Underscore (_).
5. Snake Case :

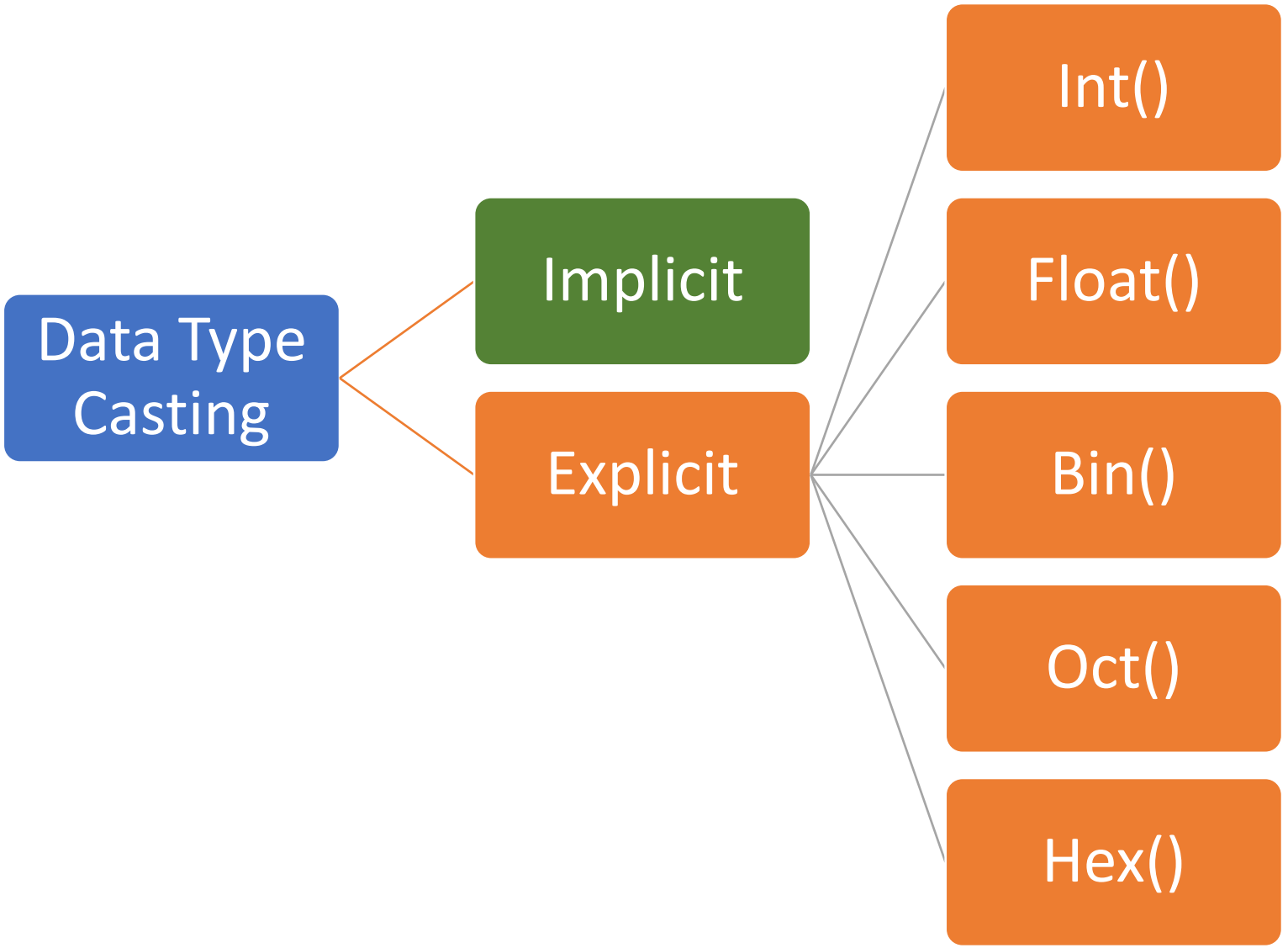
word1 word2 word3___word4

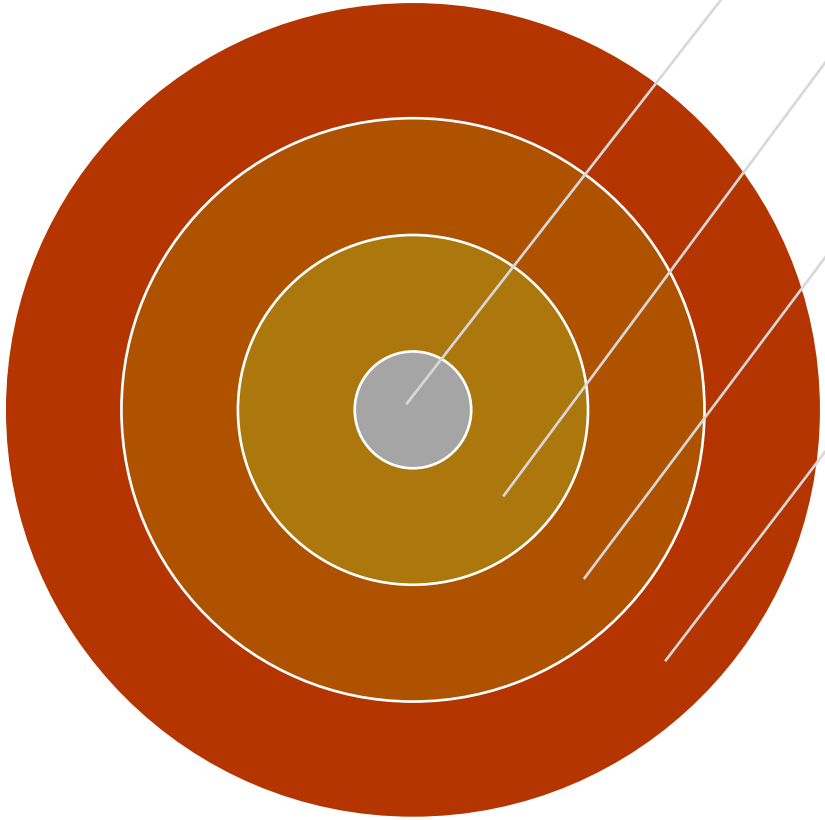

TYPES OF VARIABLE DECLARATION TYPE -1
STORING SINGLE VALUE IN SINGLE VARIABLE
TYPES OF VARIABLE DECLARATION TYPE - 2
STORING SINGLE VALUE IN MULTIPLE VARIABLES
TYPES OF VARIABLE DECLARATION TYPE - 3
STORING MULTIPLE VALUES IN MULTIPLE VARIABLES
TYPES OF VARIABLE DECLARATION TYPE-4
DONT USE SAME VARIABLE NAME FOR VALUE DECLARATION .


a b=12
Legal Variables
a=1;b=2;c=3
a b c=123


a=1
$a=1
Illegal Variables
a b =1
1a= 1

Data Conversion





What Are **Identifiers**.

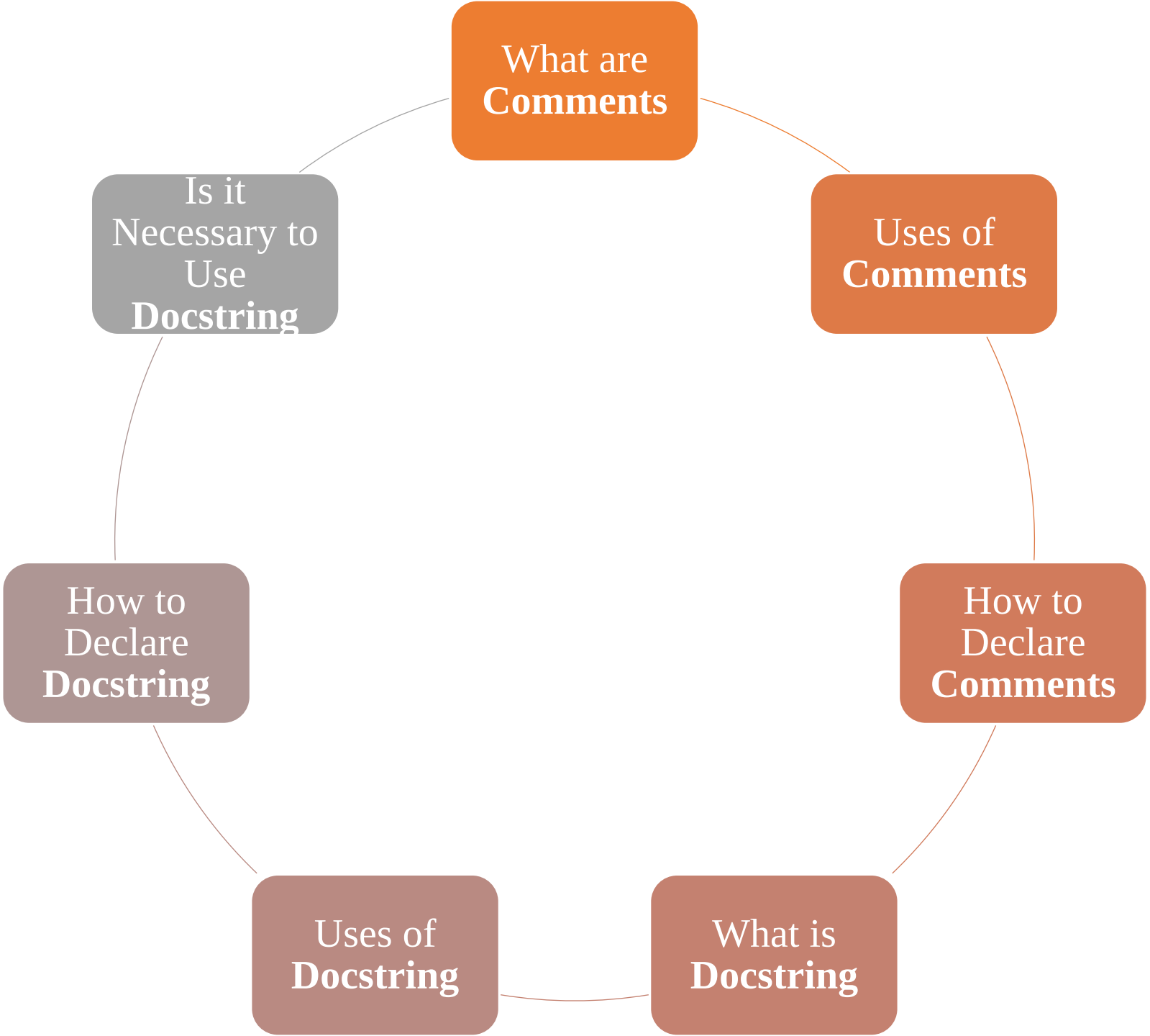
How Are **Identifiers**.

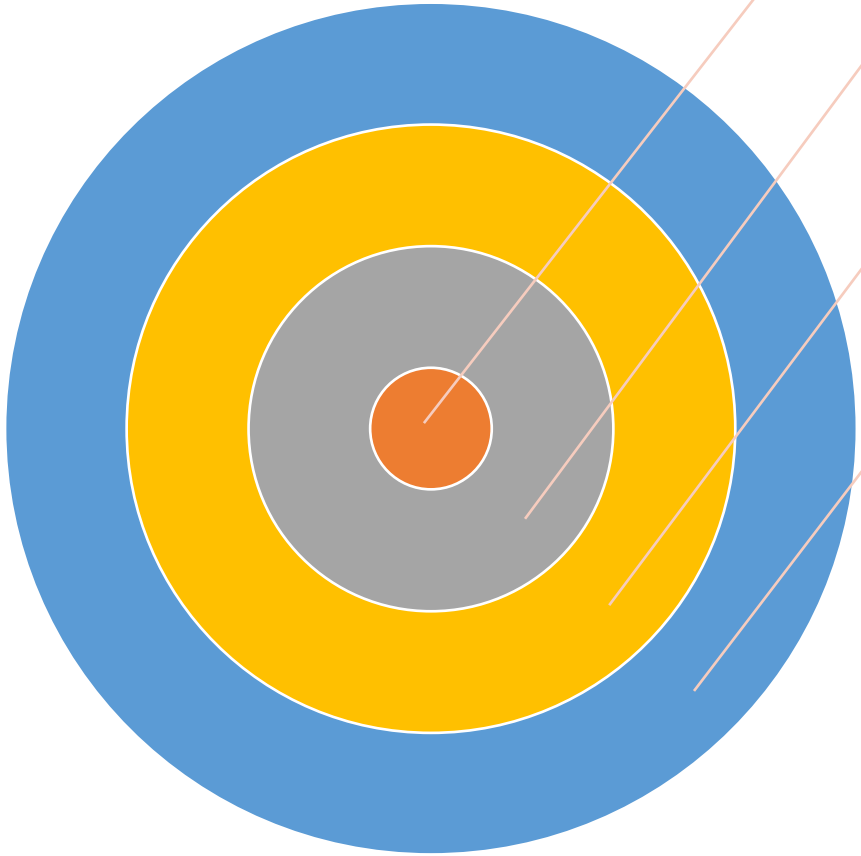
Variable Name Function Name Class Name (Or) Not?

These Are **Identifiers**?

Rules To Declare Identifiers :

- Must Starts With Letters (Upper Or Lower Case) Underscore.
- Identifiers Cannot Start With Numbers.
- Identifiers Contains Letters Underscore Numbers.
- Special Characters Are Not Allowed Except Underscore (_)
- Same Identifier Name Is Not Allowed.
- Keywords Are Not Allowed To Use As Identfer Name.
- DECLARE IDENTIFIERS IN DIFFERENT CASES :
  - CAMELCASE(collageName)
  - PASCALCASE(CollageName)
  - SNAKECASE(Collage_Name)


What are **Comments**
Uses of **Comments**
How to Declare **Comments**
What is **Docstring**
Uses of **Docstring**
How to Declare **Docstring**
Is it Necessary to Use **Docstring**



01 What is **INPUT**?

02 How to Take **INPUT**?

03 Types of **INPUT**?

04 How to store **INPUT**?

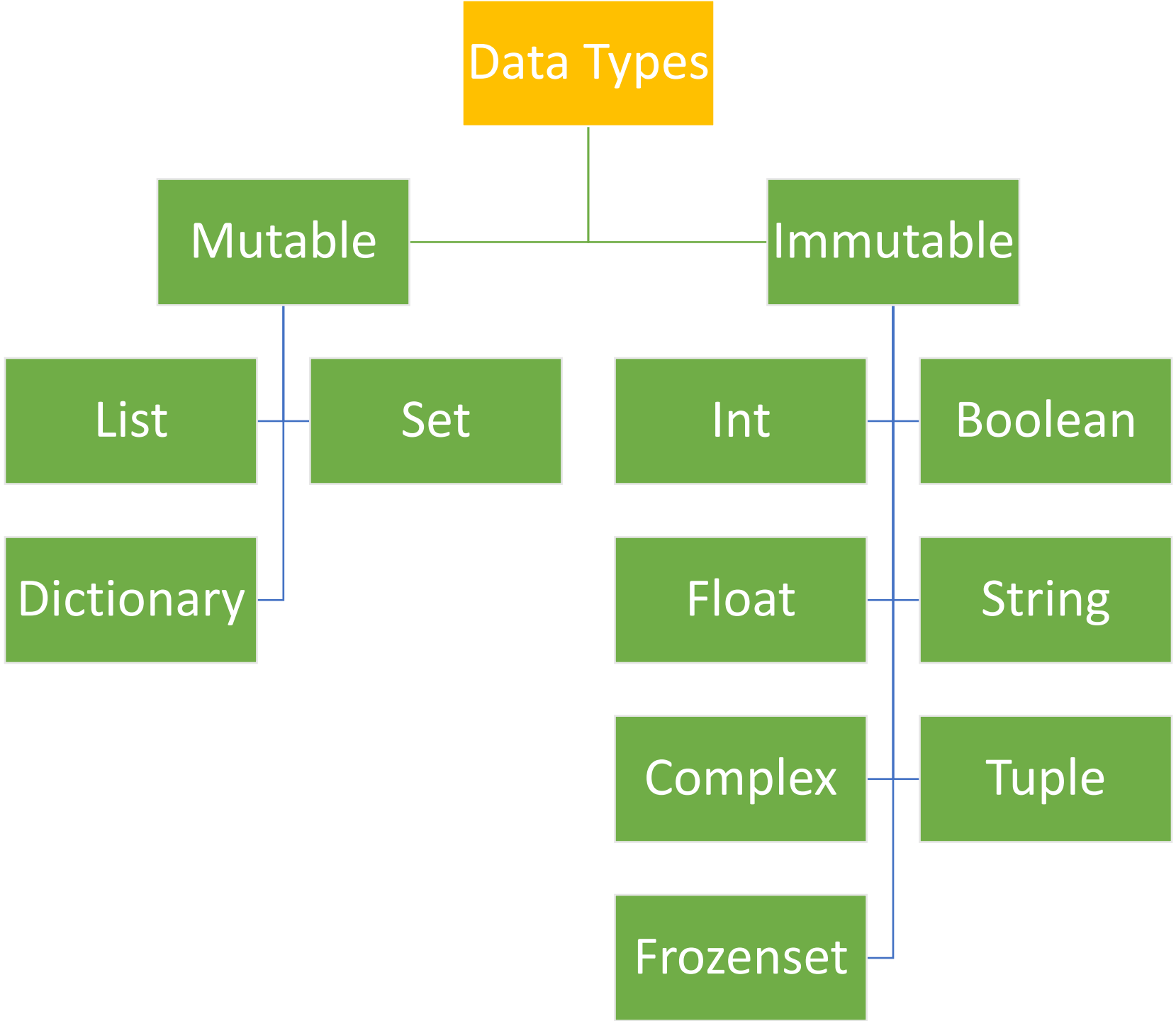
PYTHON CORE

JAVA CORE

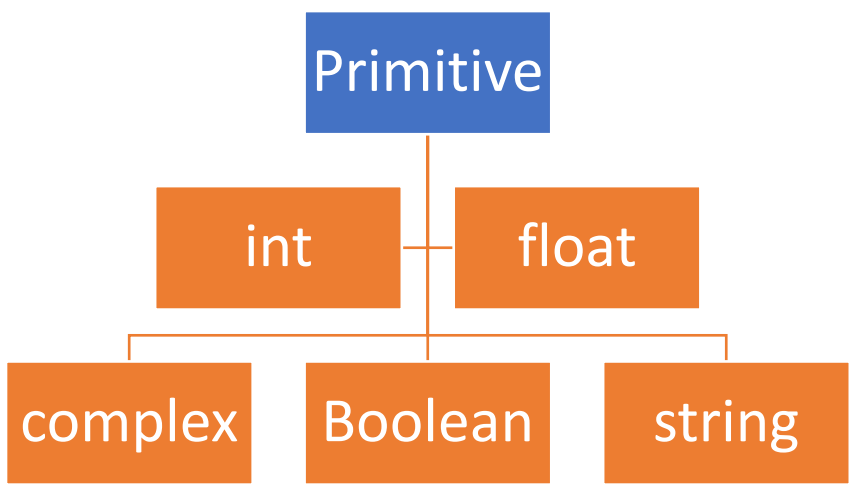
C CORE

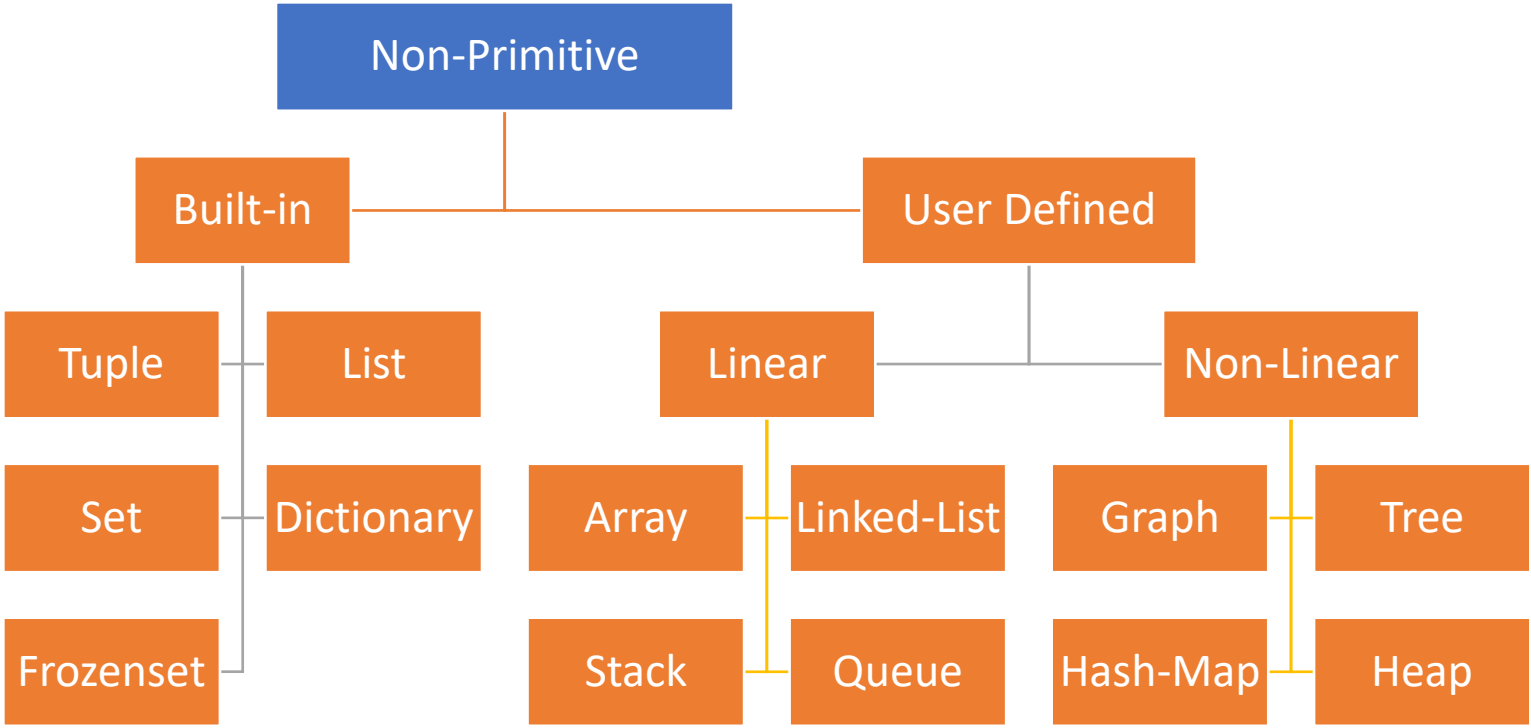
C++ CORE

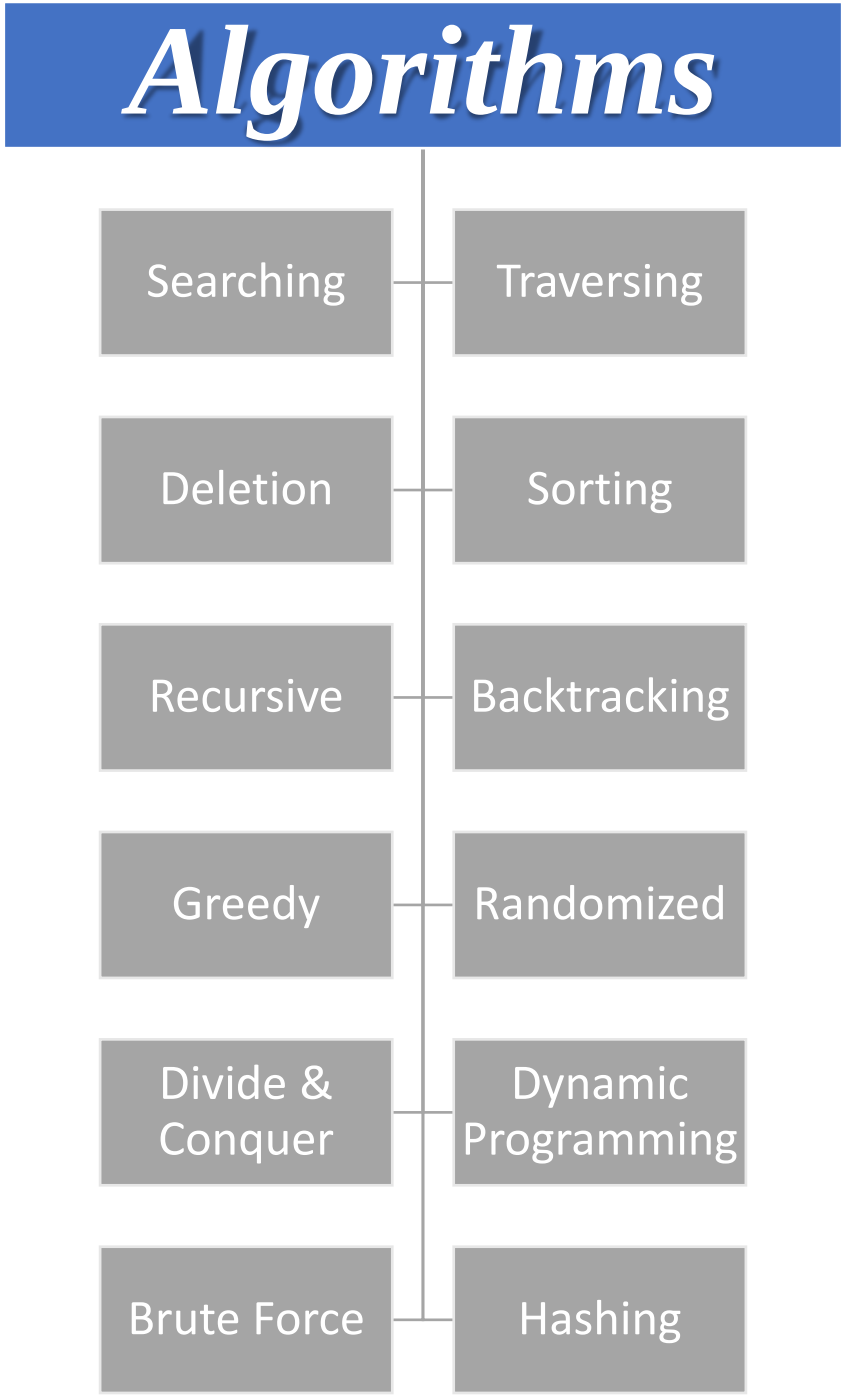
FUNDEMENTALS or BASIC


Data Types
Mutable
Immutable
List
Set
Dictionary
Int
Boolean
Float
String
Complex
Tuple
Frozenset

Data Structures






Algorithms
Searching
Traversing
Deletion
Sorting
Recursive
Backtracking
Greedy
Randomized
Divide & Conquer
Dynamic Programming
Brute Force
Hashing

KEYWORDS

def

class

True False None

it else elif

for while

break continue pass

and or not

FOR LOGICAL CONDITIONS

from import

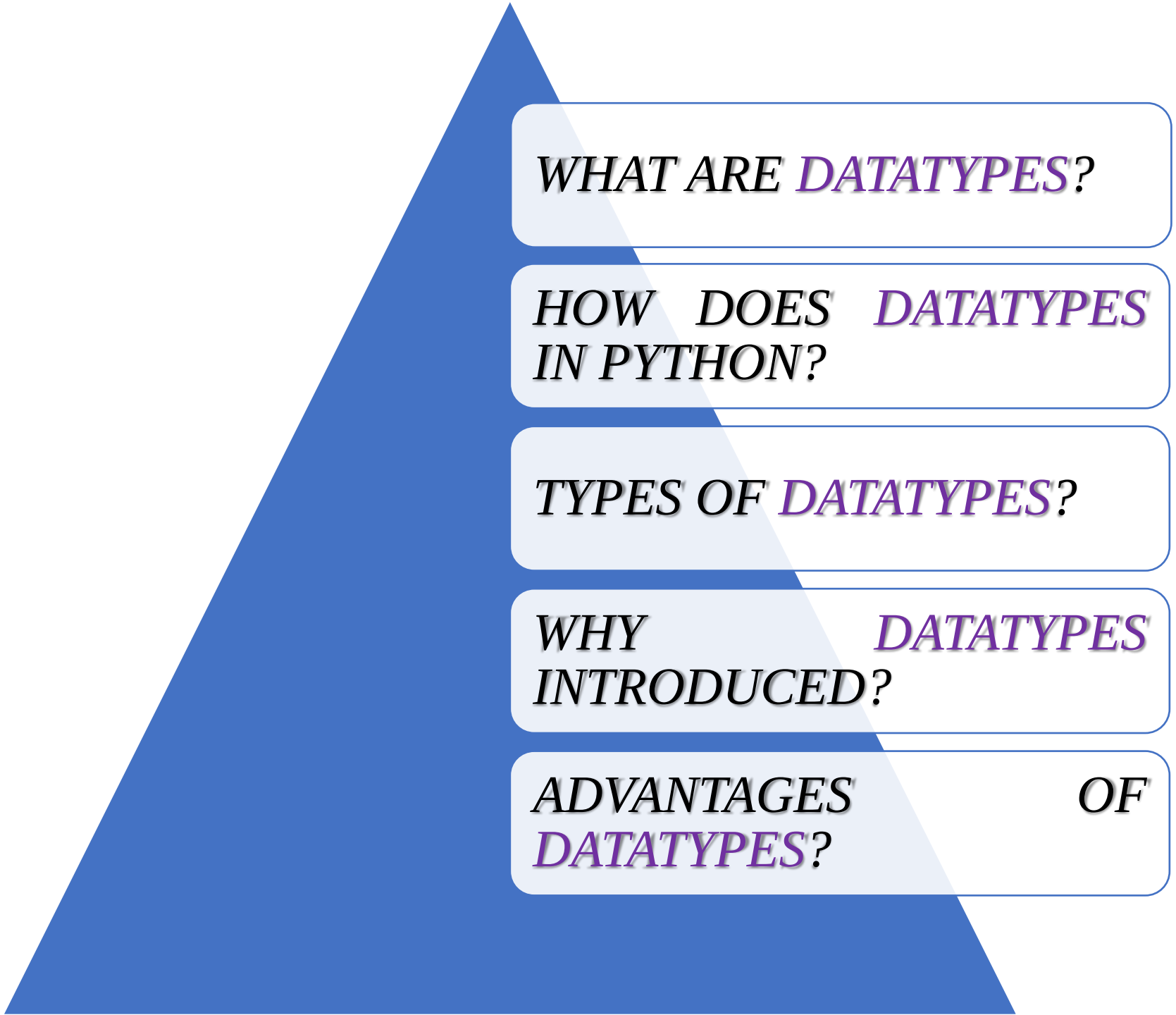
TO IMPORT MODULES

try except finally raise

EXCEPTION HANDLING

is in as lambda return
Yield global with del

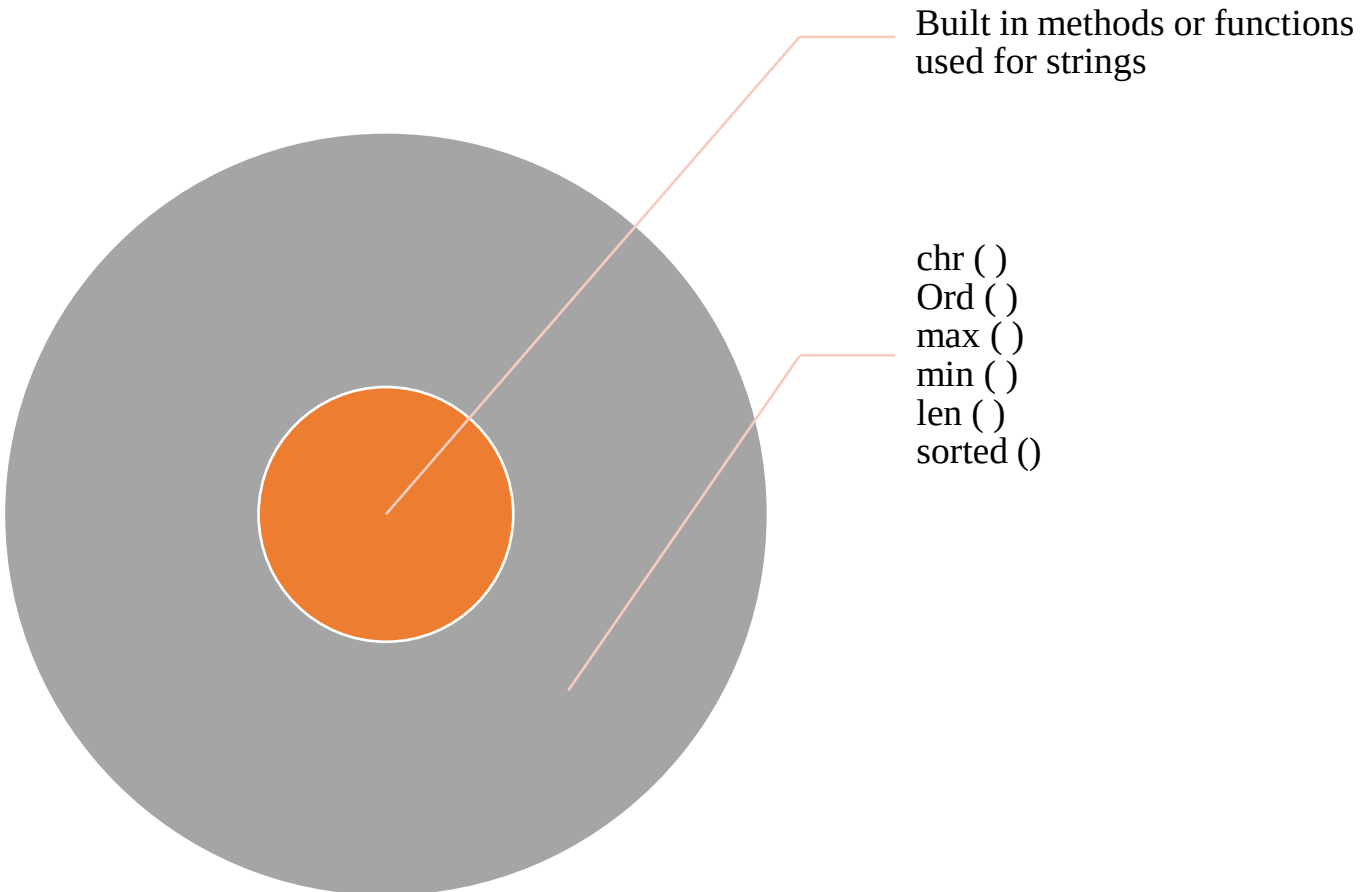
IMPORTANT OTHER


WHAT ARE DATATYPES?
HOW DOES DATATYPES IN PYTHON?
TYPES OF DATATYPES?
WHY DATATYPES INTRODUCED?
ADVANTAGES OF DATATYPES?

Python is case sensitive or not?

The Answer is ***case sensitive***

What is ASCII ?



Boolean Data Type

Boolean Data Type

These are works Control statements

- *Boolean datatype* is one of the simple datatype.
- Or logical datatype.
- Or conditional datatype.

String Data Type

Python String Methods

'capitalize'	'casefold'	'center'	'count'	'encode'	'endswith'	'expandtabs'	'find'
'format'	'format_map'	'index'	'isalnum'	'isalpha'	'isascii'	'isdecimal'	'isdigit'
'isidentifier'	'islower'	'isnumeric'	'isprintable'	'isspace'	'istitle'	'isupper'	'join'
'ljust'	'lower'	'lstrip'	'maketrans'	'partition'	'removeprefix'	'removesuffix'	'replace'
'rfind'	'rindex'	'rjust'	'rpartition'	'rsplit'	'rstrip'	'split'	'splitlines'
'startswith'	'strip'	'swapcase'	'title'	'translate'	'upper'	'zfill'	

Most important Strings Methods

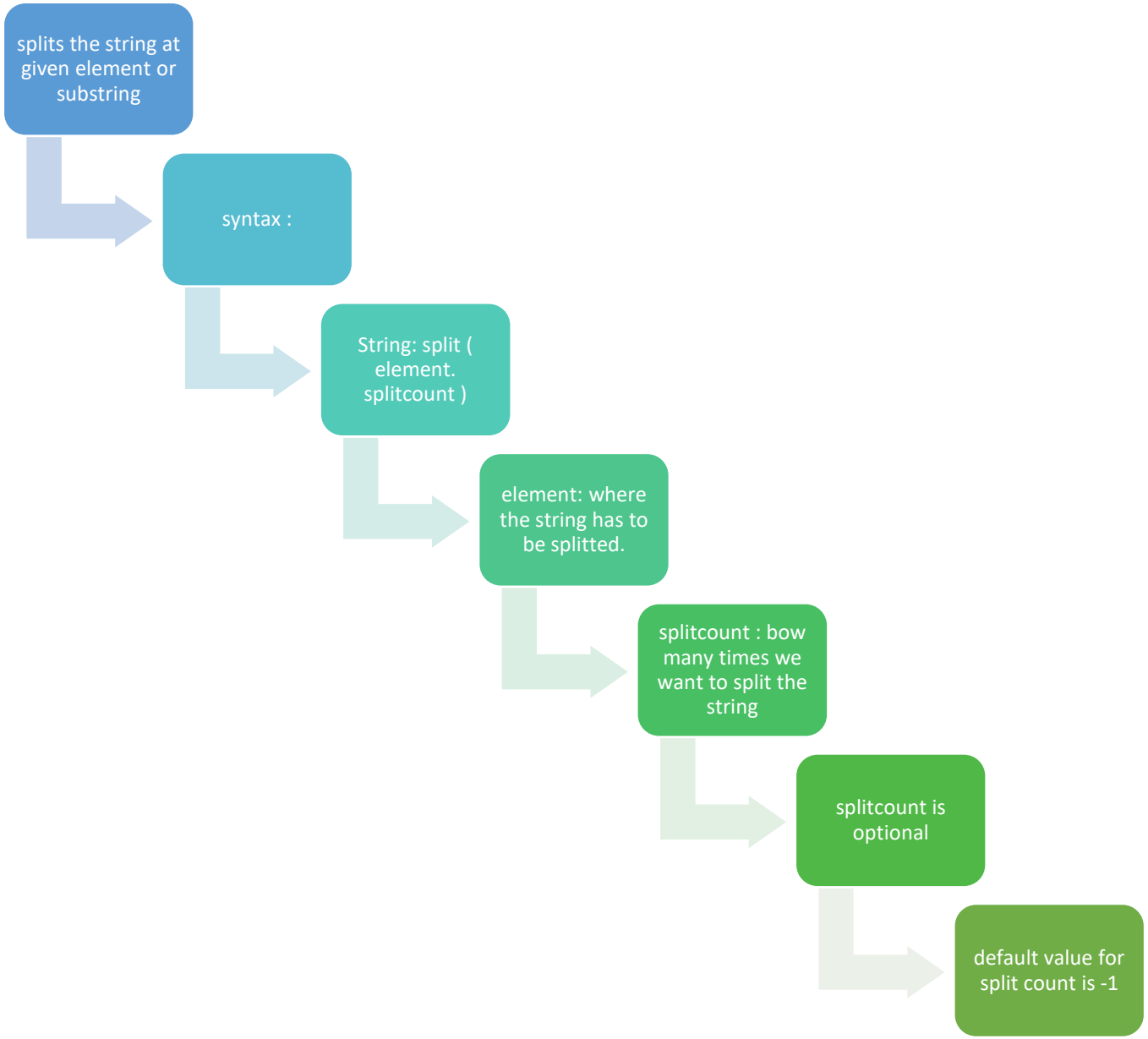
- capitalize
- title
- istitle
- upper
- isupper
- lower
- islower
- swapcase
- index(element, start, end)
- find (element, start, end)
- join( )
- split( )
- Strip()
- Isspace()
- replace(old new count)
- Count(element start end)
- startswith(parameter)
- endswith(parameter)
- isidentifier(parameter)
- isalpha
- isdecimal
- isdigit(subscript, superscript)
- isnumeric
- isalnum= isalpha() + isnumeric()
- splitlines
- format

String Indexing & Slicing In Python

What Is String Indexing
How To Do String Indexing Using Positive And Negative Index ?
What Is String Slicing ?
How To Do String Slicing Using Positive Index ?

string [start: end: step]

Split


splits the string at given element or substring
syntax :
String: split (element. splitcount)
element: where the string has to be splitted.
splitcount : bow many times we want to split the string
splitcount is optional
default value for split count is -1

Join

join method joins the string with iterators

SYNTAX:
string.join(iterator)

Isnumeric()

isnumeric() method:

This method is available for string objects and checks if all characters in the string are numeric characters (0-9).

It returns True if all characters in the string are numeric otherwise it returns False.

Example: my_string = "12345“

result = my_string.isnumeric()

print(result)

True

isdecimal()

isdecimal() method:

Similar to isnumeric() this method is available for string objects and checks if all characters in the string are decimal characters (0-9).

It returns True if all characters in the string are decimal otherwise it returns False.

Example: my_string = "12345“

result = my_string.isdecimal()

print(result)

True

isdigit()

isdigit() method:

This method is also available for string objects and checks if all characters in the string are digits (0-9).

It returns True if all characters in the string are digits otherwise it returns False.

Example: my_string = "12345“

result = my_string.isdigit()

print(result)

True

String Formatting In Python

.format() is a string method that helps us format our string Using curly braces ({}) as a placeholder. Let’s understand its syntax with the help of an example.

.format() method

Name= ”joy”

Curly Braces Act as Placeholder

age = 22

print (“my name is { } and I am { } years Old" .format (name age))

Output: My name is Doy and I am 22 years old

variable name for

Placeholder

The .format() method is very helpful for formatting a string. However it has one drawback it is not very readable. That's why in Python 3.6 a new way to format strings was introduced known as f -strings.

f-strings

Write variables inside curly braces

print(f”name is {name} and I am (age) years old")

Write" Before your String to indicate that it is f-string.

List Data Type

List Datatype in Python

We Can Store Any Datatype In The List.

To Handle Multiple Values Having Any Datatype At A Time.

How To Declare List ?

List Enclosed With Square Brackets [].

Values Are Separated By Using Comma ( ) In List.

Why List Are Compound Datatype?

If Multiple Values Are Stored In Single Datatype Then It Is Compound Datatype .

Why Lists Are Heterogenous Datatype ?

Using Different Datatypes At A Time In Single Datatype Is Called Heterogenous Datatype.

Why Lists Are Sequence Datatypes?

Python List Methods

- append
- copy
- clear
- count
- extend
- insert
- index
- pop
- remove
- reverse
- sort
- min
- max
- updation

Slicing and indexing

main list :

['a', 1, 'b', 2, 'c', 3, 'd', 4]

negative index	-8	-7	-6	-5	-4	-3	-2	-1
positive index	0	1	2	3	4	5	6	7

start = -8

sublist :

['a', 'b', 'c', 'd']

Slicing for rules

1. if you want to give start value : 0 tor slicing no need to give because default value for start value is 0.

2. if you want to give end value length of list in slicing no need to give because default value for end value is length of list.

3. if you want to give start value : 0 and end value length of list no need to give because default value for start value is 0 and end value is length of list.

4. don't use positive value and negative value as start and end values in a string slicing query.

Advantages of lists

lists store data in sequential order.

so we can perform indexing and slicing.

Slicing is accessing multiple values/ sublist from a list.

Set Data Type

Set datatype in python

Use set when we don't want duplicate values in our data.

If we don't want to store the data in order Set is better.

Time complexity will be efficient.

Set doesn't store data in sequence.

Set is a heterogenous compound collection mutable DataStucture.

Set is a Compound DataStucture ?

Collection data structures are Two Types :

1. Set

2 .Dictionary

The data in set must enclose with curly braces { } & every element in set is separated by comma.

Heterogenous Datatype can store different datatypes at a time in single datatype.
Set stores immutable datatypes.
Set doesn't store - list dictionary array. because these are mutable datatypes.
Set doesn't support Indexing & Slicing because It is not a sequence datatype.
Set is a collection datatype so it doesn't store data in order so Indexing & slicing is not supported.
Sets are mutable datatypes.
Heterogenous Datatype can store different datatypes at a time in single datatype.

Set user input

syntax: **set (input ( ) .split( ))**

Set user input int

To Take set Values As Integers: **set (map (int input ( ) .split())**

Python Set Methods

- add
- clear
- copy
- difference
- difference_update
- discard
- intersection
- intersection_update
- isdisjoint
- issubset
- issuperset
- pop
- remove
- symmetric_difference
- symmetric_difference_update
- union
- update

Tuple Data Type

Tuple datatype in python

Tuples must enclose with parenthesis ( ) & elements in tuple must separate with comma.

Tuple is Immutable Datatype.

Syntax: (value1 value2).

To create Tuple: variable = ( ) or tuple ( ).

Tuple is a sequence datatype so it supports slicing.

Slicing in Tuple is same as String & List

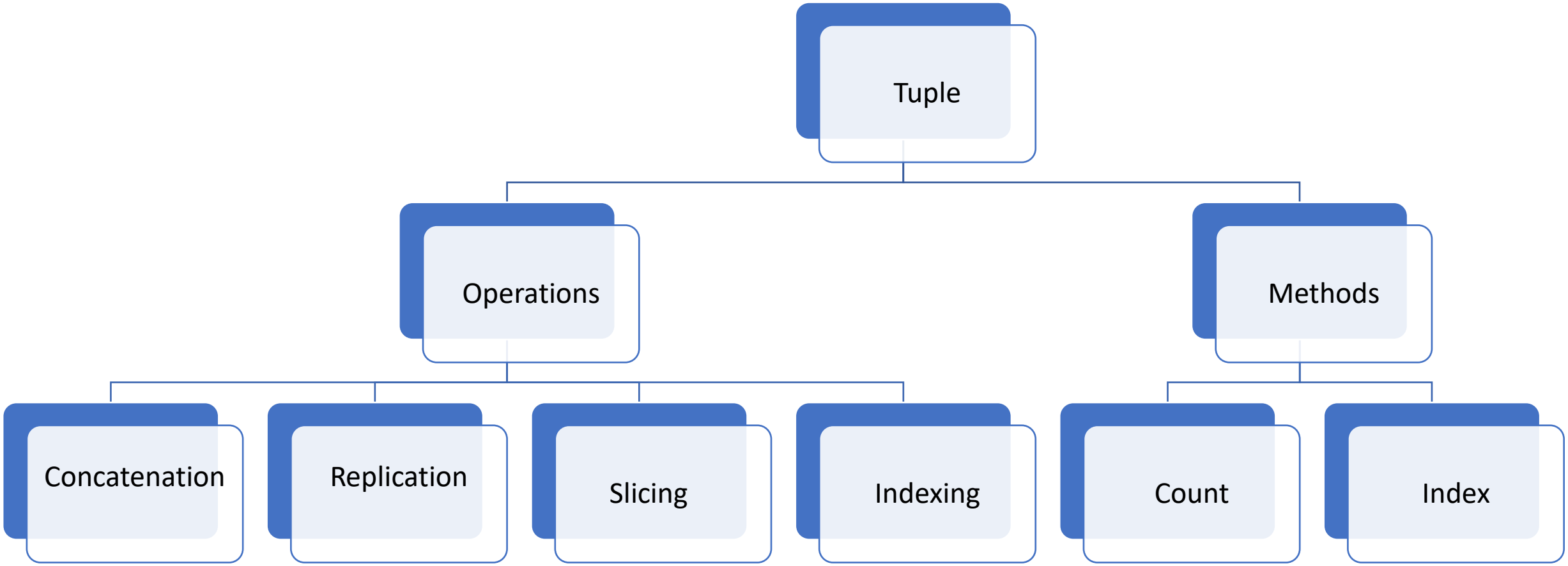
Syntax: variablename[start: stop: step]

Tuple only two methods.

Tuple contains Only 2 methods.

1. count ( )
2. index ( )

Python Tuple Methods and Operations



How to take input for tuples

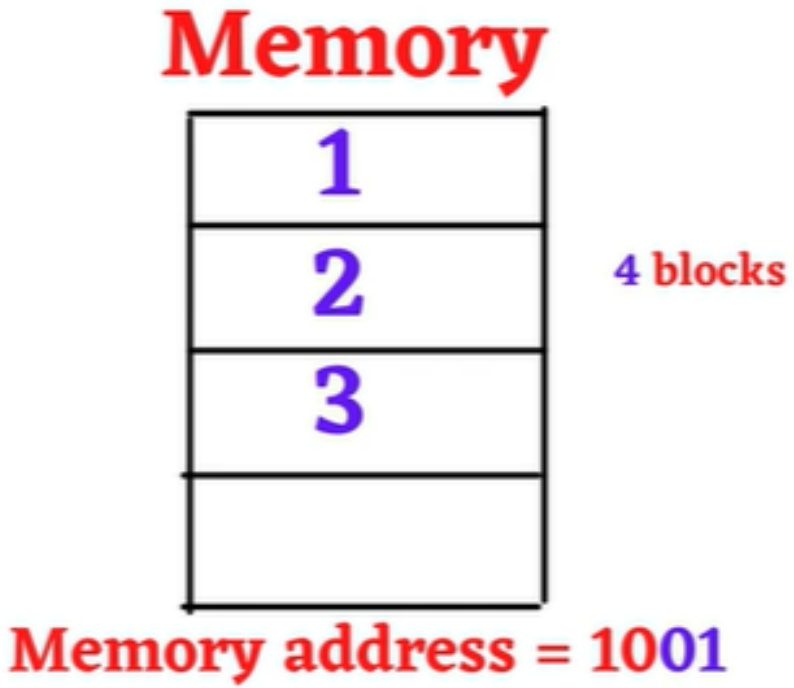
Syntax: tuple(input().split())

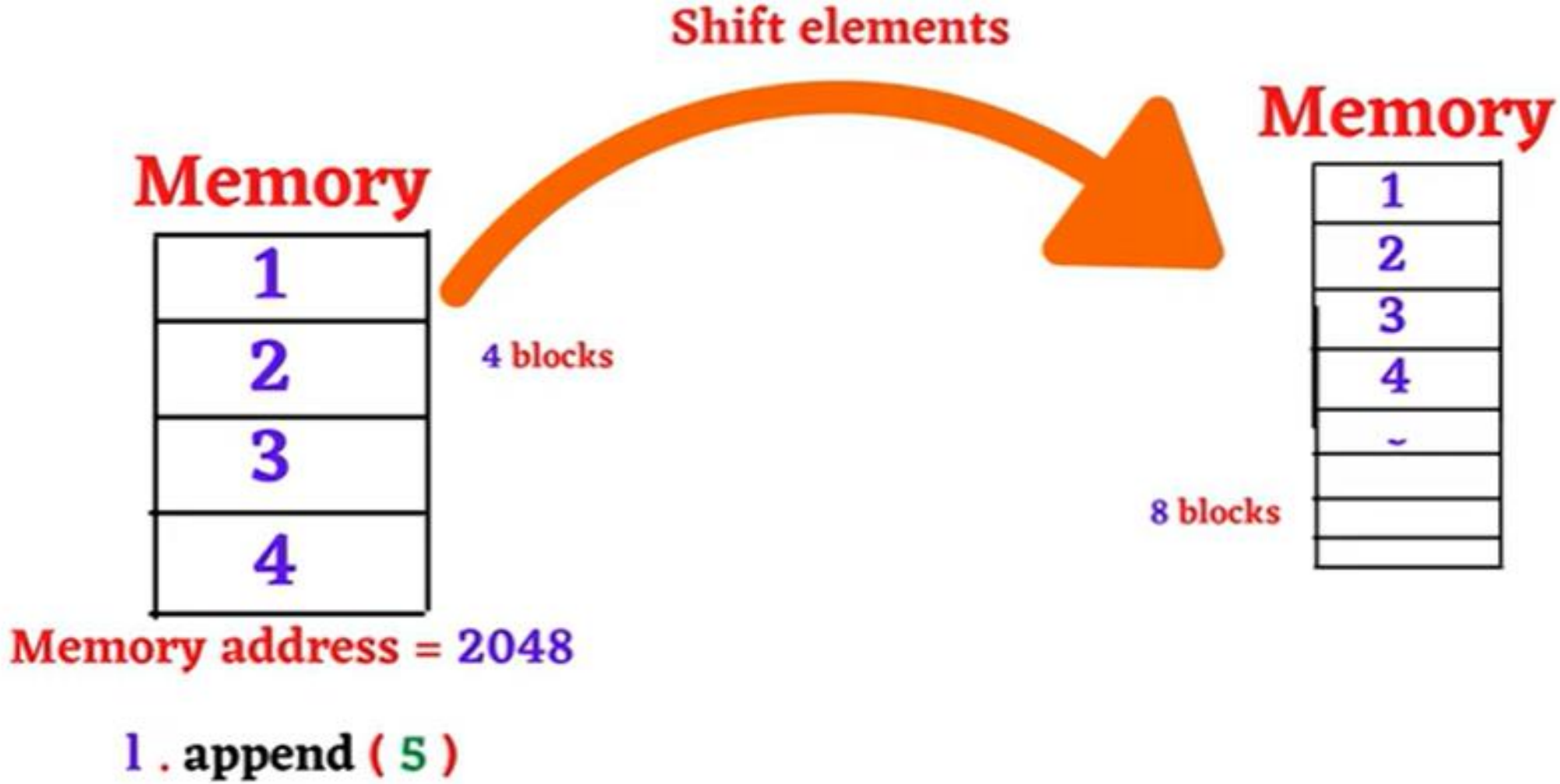
To Take Tuple As Integers :

Syntax: tuple (map (int input ( ) .split())

why tuples faster than list ?

l = [1, 2 , 3]




Shift elements
Memory
1
2
3
4
4 blocks
Memory
1
2
3
4
8 blocks
Memory address = 2048

l . append (5)

Dictionary Data Type

Dictionary Datatype in Python

dictionary datatype is a compound heterogenous mutable collection DataStucture .

data in dictionary stores in key value pair.

Accessing value by using Key.

In a dictionary a key has so many values.

dictionary doesn't allow duplicate keys.

Dictionary DataStucture stores data same as dictionary.

dictionary DataStucture doesn't store data in sequence like set So it is collection datatype.

we can store any datatype in dictionary like int float tuple list set etc... so it is heterogenous datatype.

dictionary DataStucture is a mutable so we can update add & remove items in dictionary.

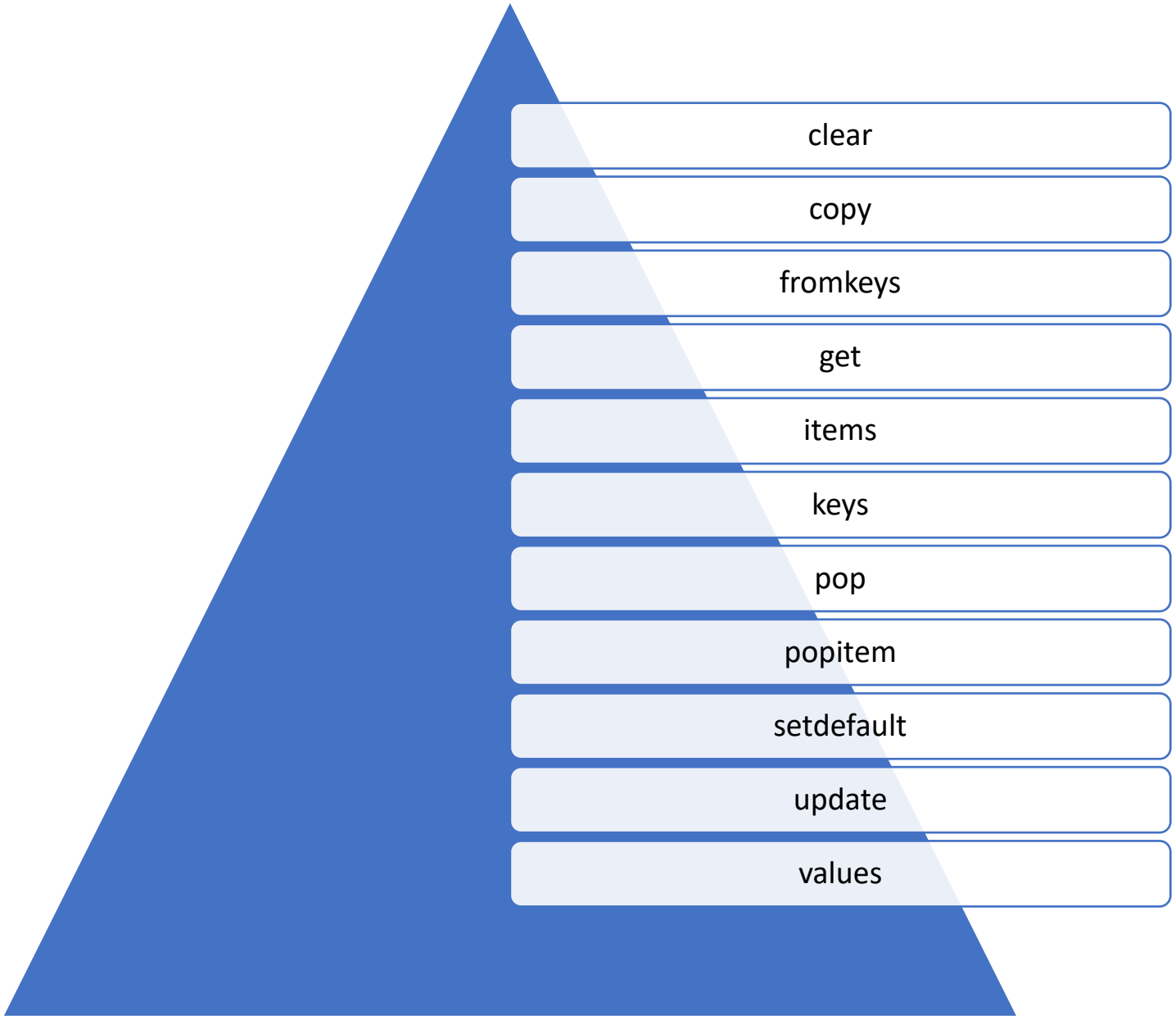
How to declare a dictionary ?
{ }---> empty closed curly braces.
dict ( ) function.
A dictionary can have multiple key value
pairs & enclosed with curly braces.
Every key and value separated with colon (:)
Every key value pair is separated with comma ( ) A dictionary can have multiple key value pairs & enclosed with curly braces.
Every key and value separated with colon (:)
Every key value pair is separated with comma ( )

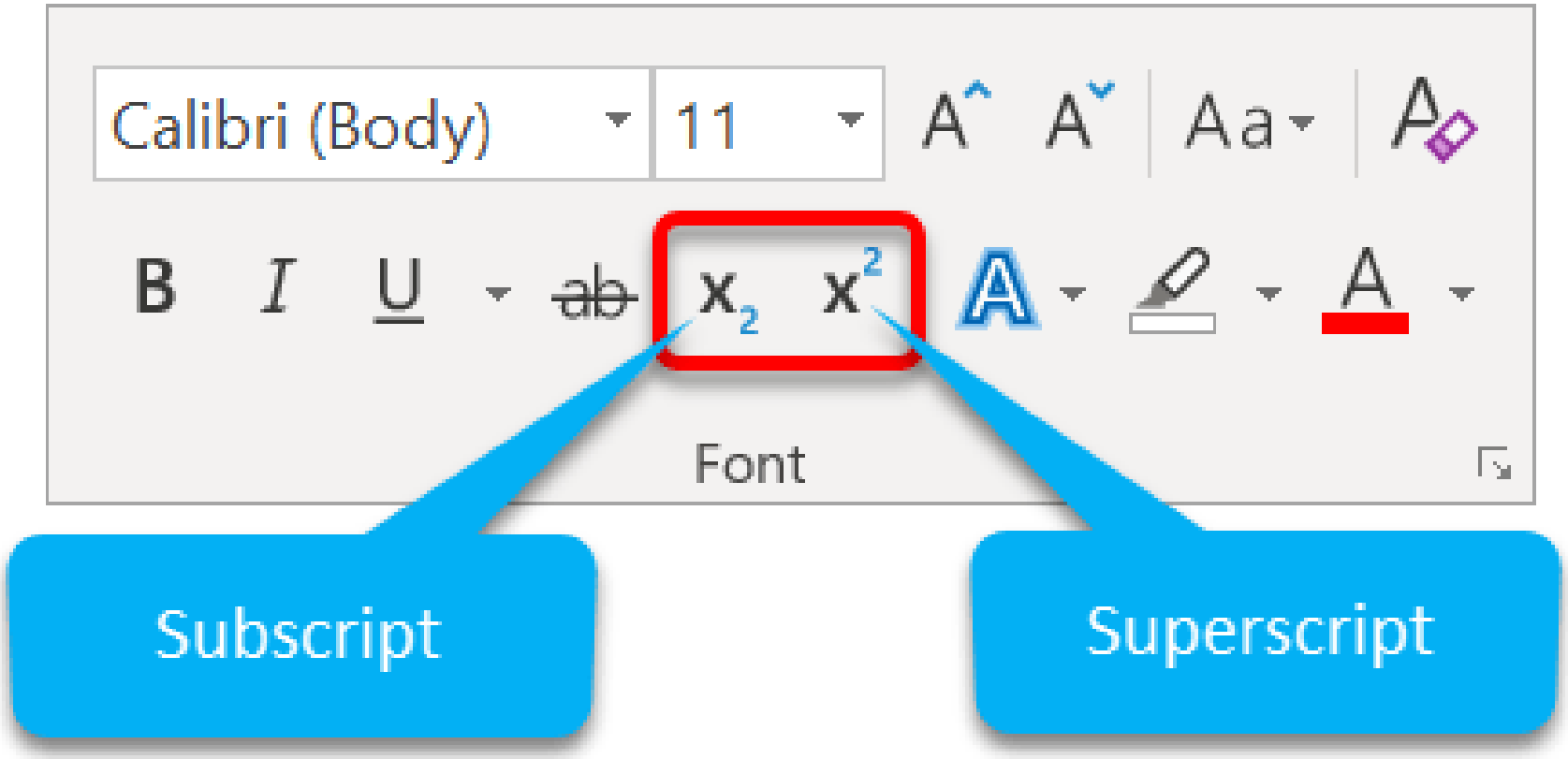
Syntax : { keyl: value1 key2: value2 key3: value3.....keyN: valueN}

(key value) ---> item

dictionary has numbers strings tuples datatypes as keys.
keys supports only immutable datatype.
Keys in dictionary doesn't repeat if keys are repeated old value in key will update with new value.

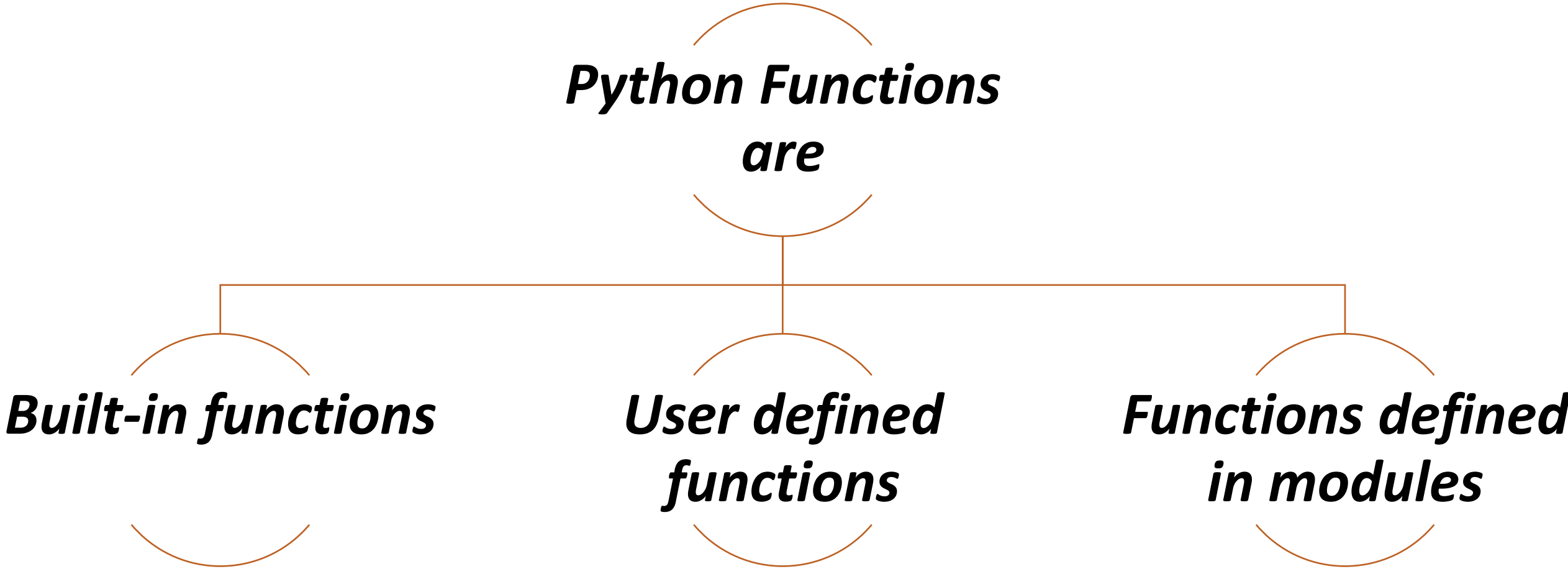
Python Dictionary Methods




Calibri (Body)
11
Aa
B
I
U
ab
x2
x2
Font
Subscript
Superscript

Python Functions

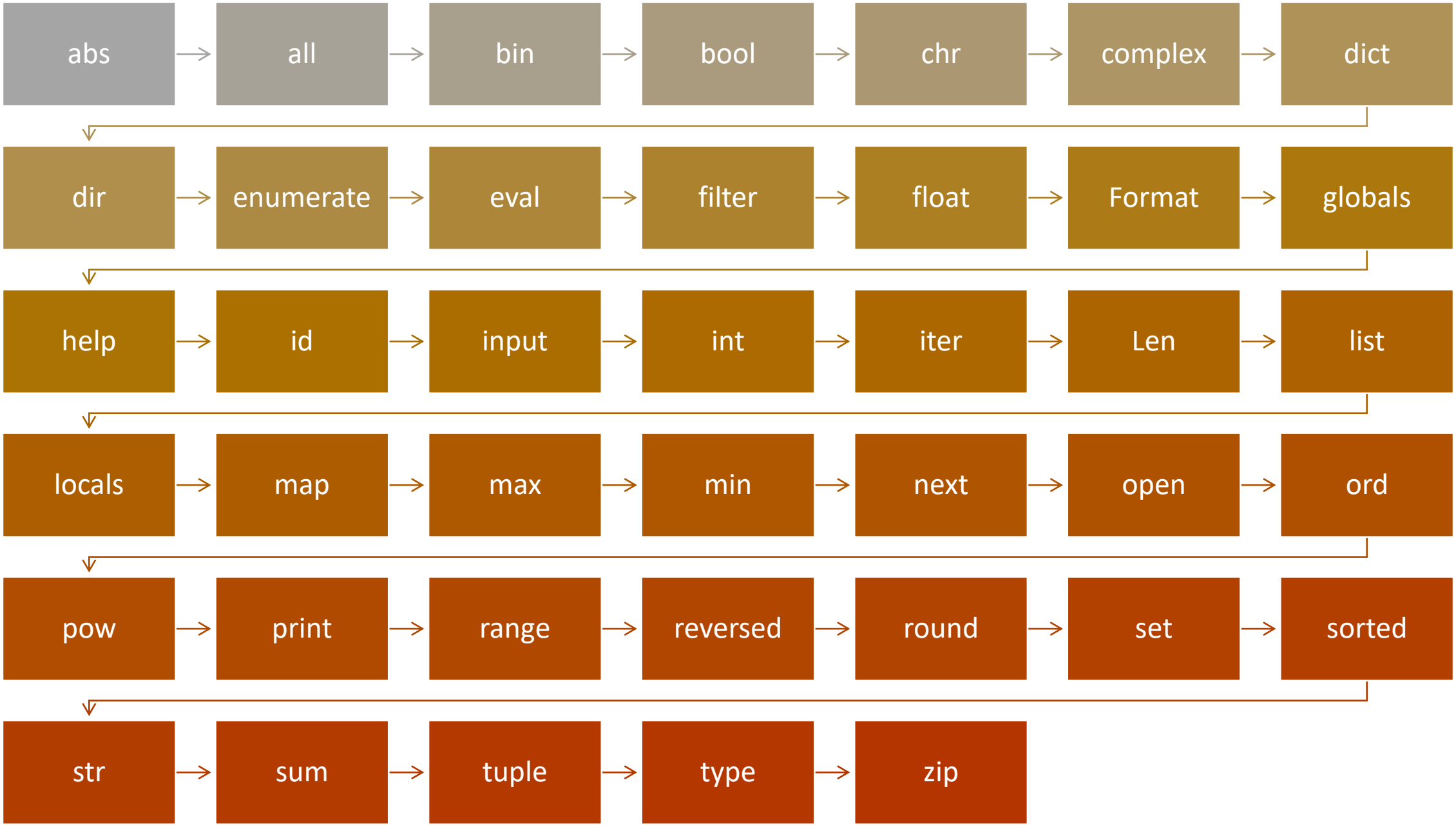
Python Functions



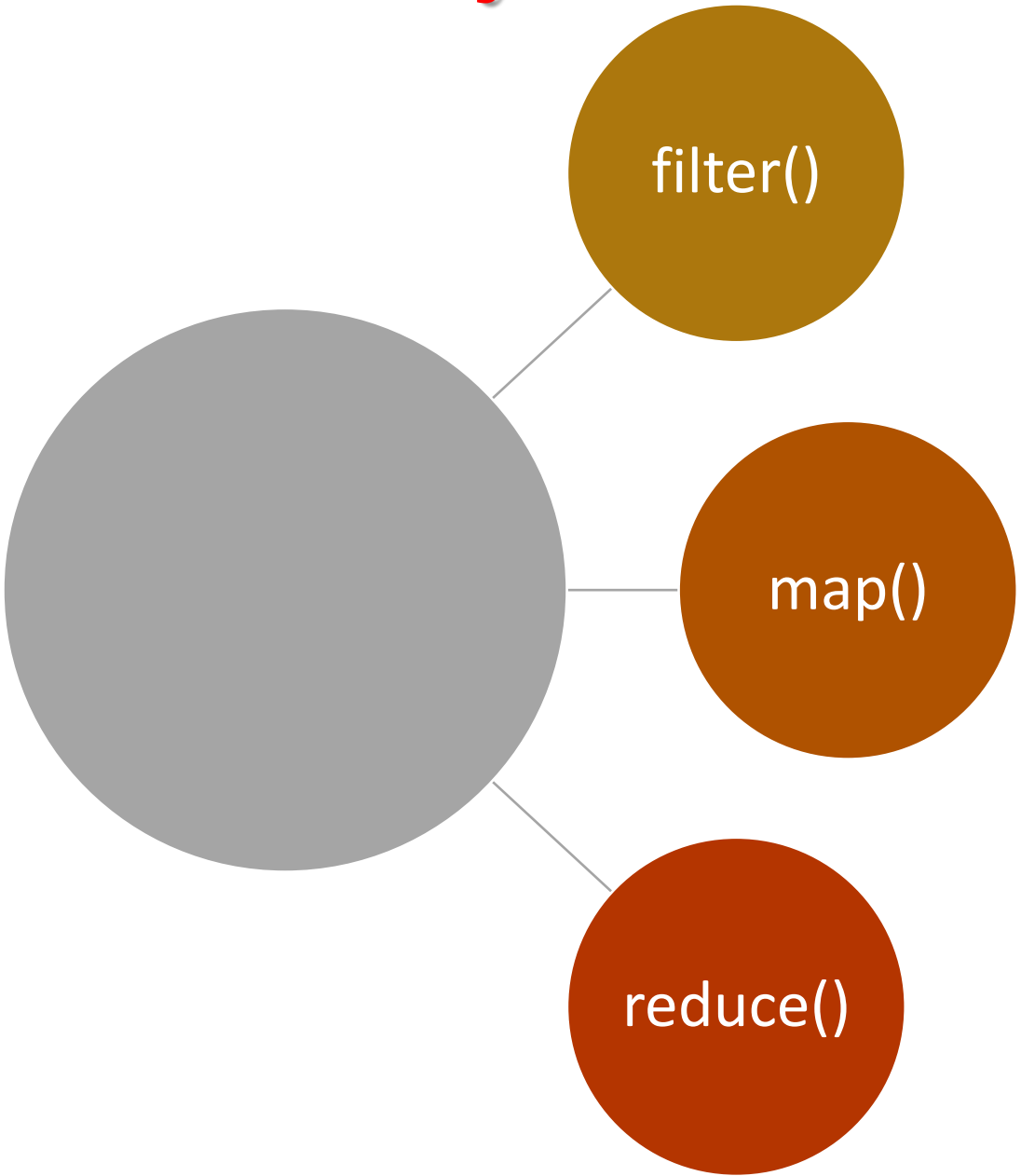
Python Built in Functions

Python Built in Functions

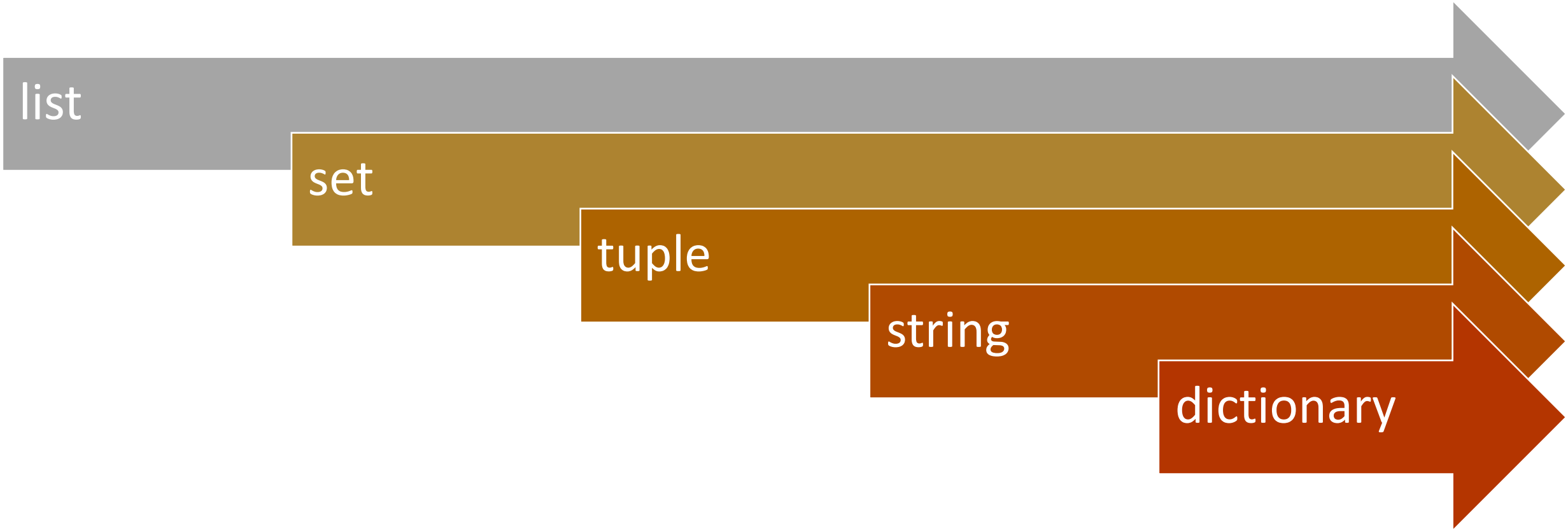
Most Important Python Built-in Functions



As a data scientist you must have these functions

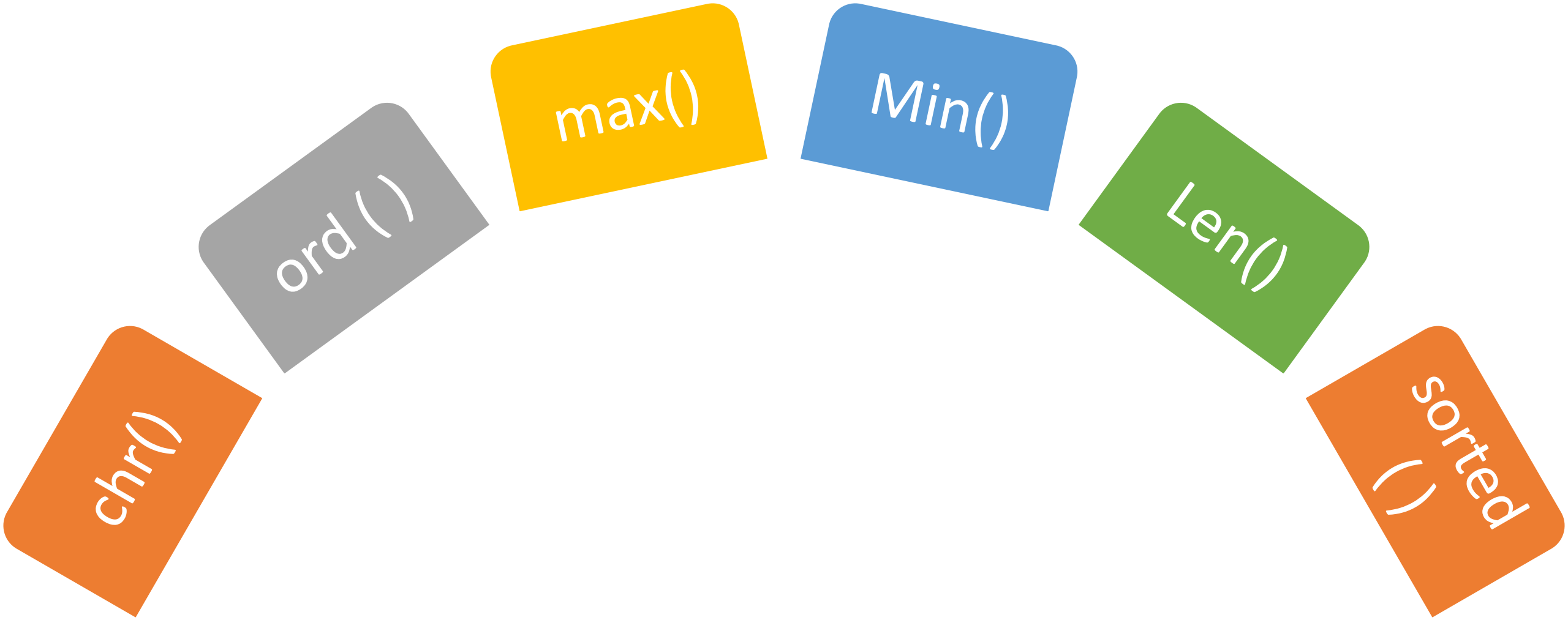


Built In Functions Or Methods Used For Strings



ASCII Table

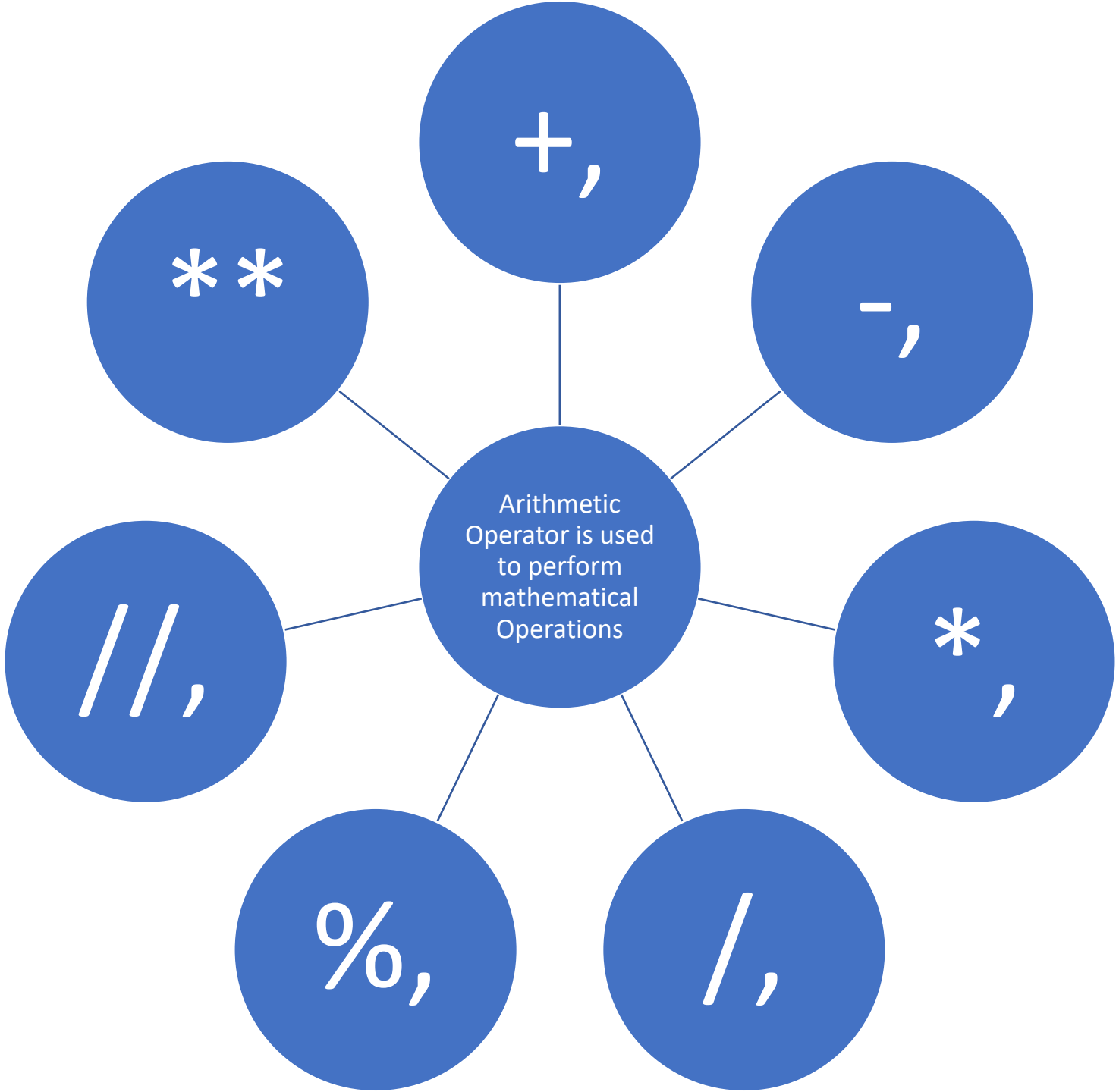
0	NUL	16	DLE	32		48	0	64	@	80	P	96	`	112	p
1	SOH	17	DC1	33	!	49	1	65	A	81	Q	97	a	113	q
2	STX	18	DC2	34	"	50	2	66	B	82	R	98	b	114	r
3	ETX	19	DC3	35	#	51	3	67	C	83	S	99	c	115	s
4	EOT	20	DC4	36	$	52	4	68	D	84	T	100	d	116	t
5	ENQ	21	NAK	37	%	53	5	69	E	85	U	101	e	117	u
6	ACK	22	SYN	38	&	54	6	70	F	86	V	102	f	118	v
7	BEL	23	ETB	39	'	55	7	71	G	87	W	103	g	119	w
8	BS	24	CAN	40	(	56	8	72	H	88	X	104	h	120	x
9	HT	25	EM	41	)	57	9	73	I	89	Y	105	i	121	y
10	LF	26	SUB	42	*	58	:	74	J	90	Z	106	j	122	z
11	VT	27	ESC	43	+	59	;	75	K	91	[	107	k	123	{
12	FF	28	FS	44	,	60	<	76	L	92	\	108	l	124	|
13	CR	29	GS	45	-	61	=	77	M	93	]	109	m	125	}
14	SO	30	RS	46	.	62	>	78	N	94	^	110	n	126	~
15	SI	31	US	47	/	63	?	79	O	95	_	111	o	127	DEL


chr()
ord ()
max()
Min()
Len()
sorted ()

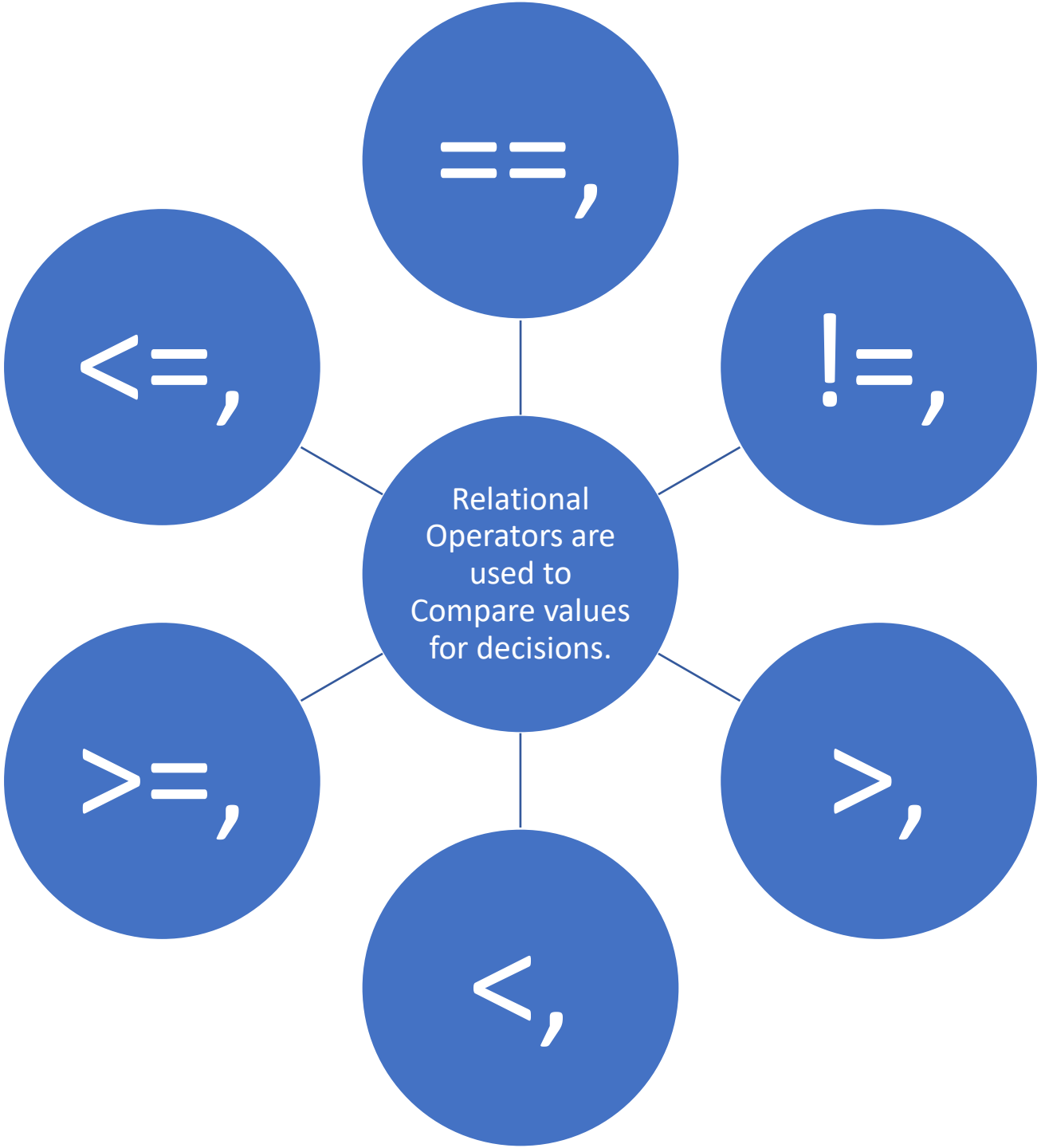
Python Built in Functions

Operators in Python

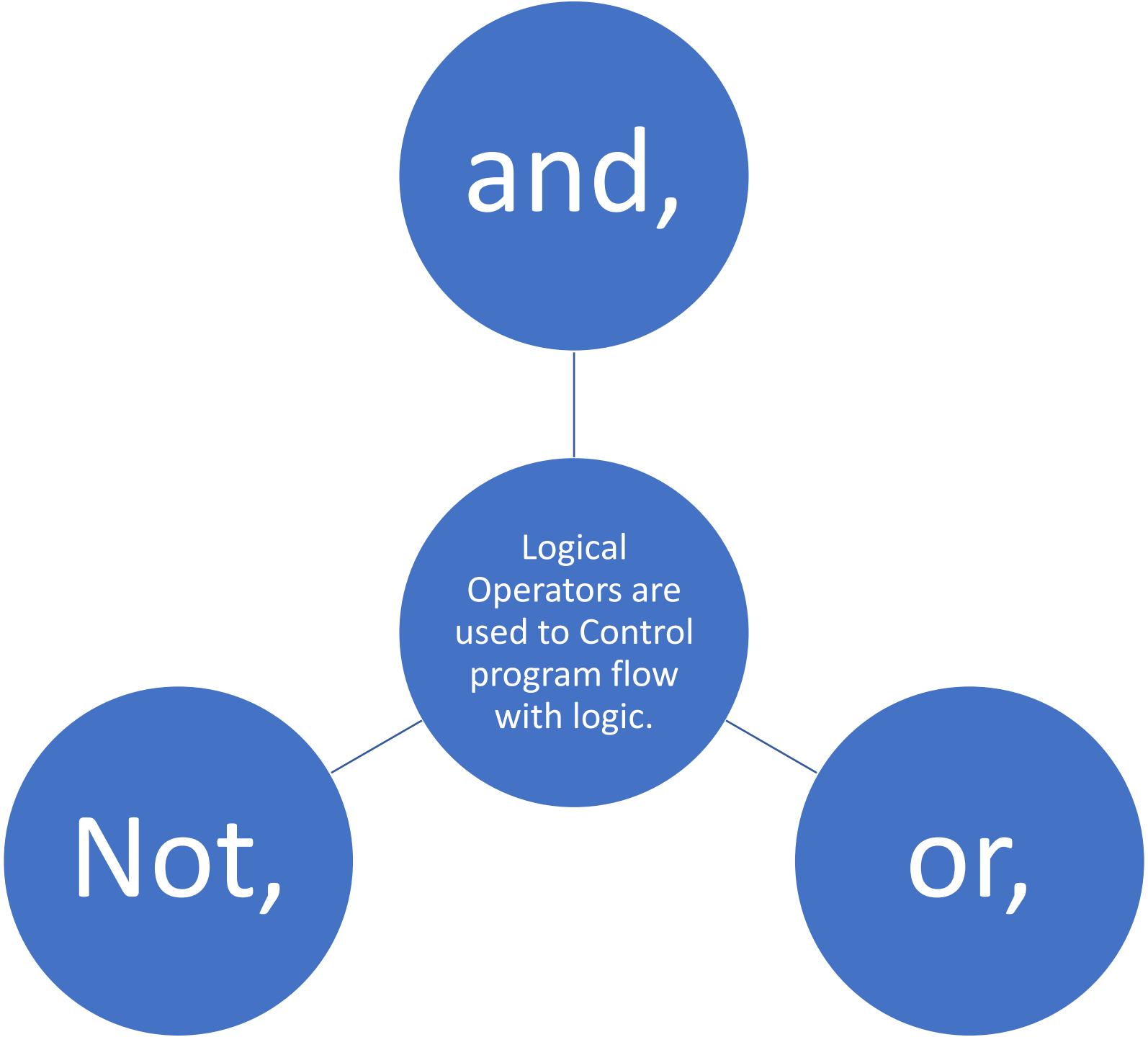
Arithmetic Operators


+,
**
-,
Arithmetic Operator is used to perform mathematical Operations
//,
*,
%,
/,

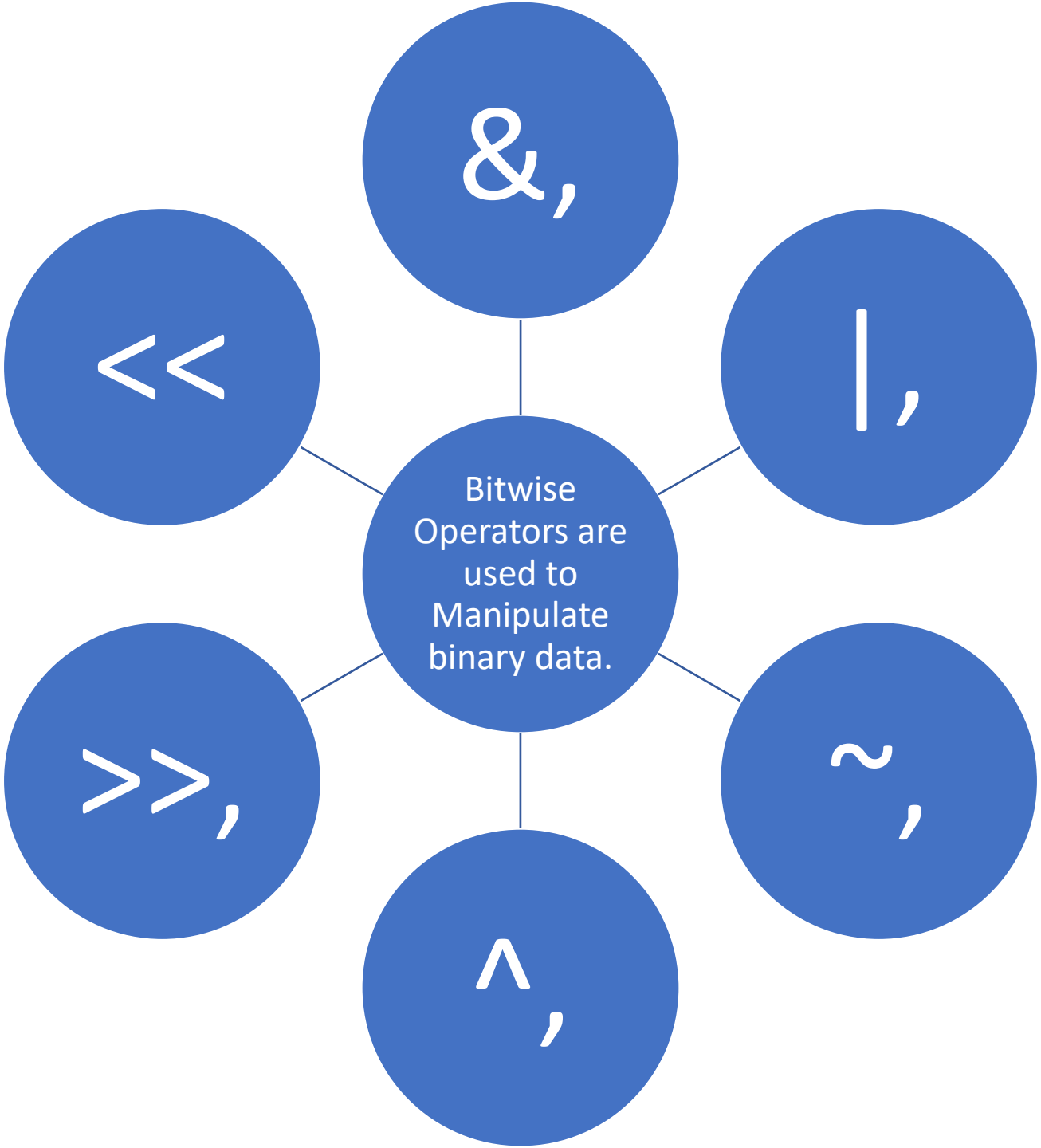
Relational Operators


==,
<=,
!=,
Relational Operators are used to Compare values for decisions.
>=,
>,
<,

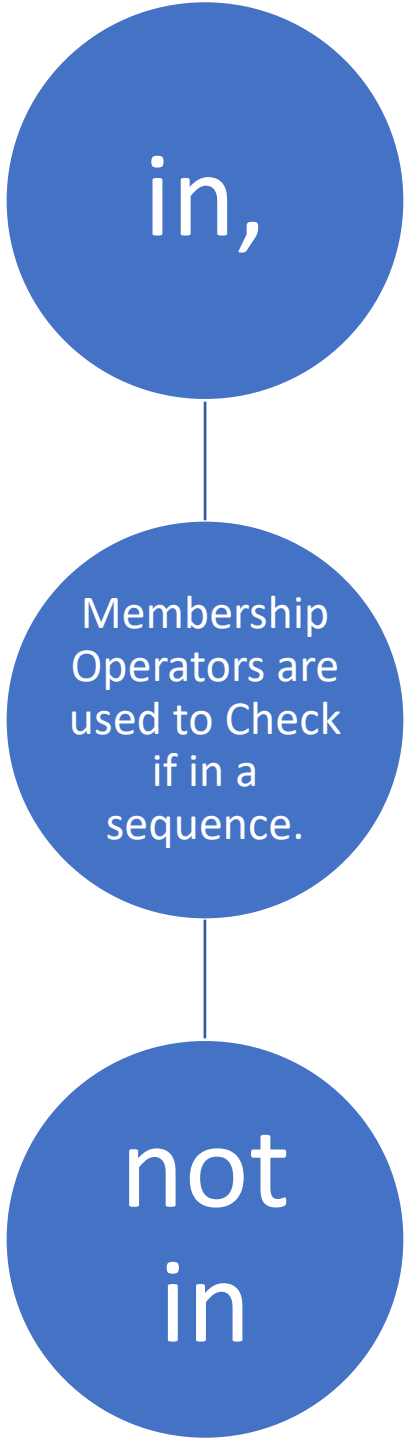
Logical Operators


and,
Logical Operators are used to Control program flow with logic.
Not,
or,

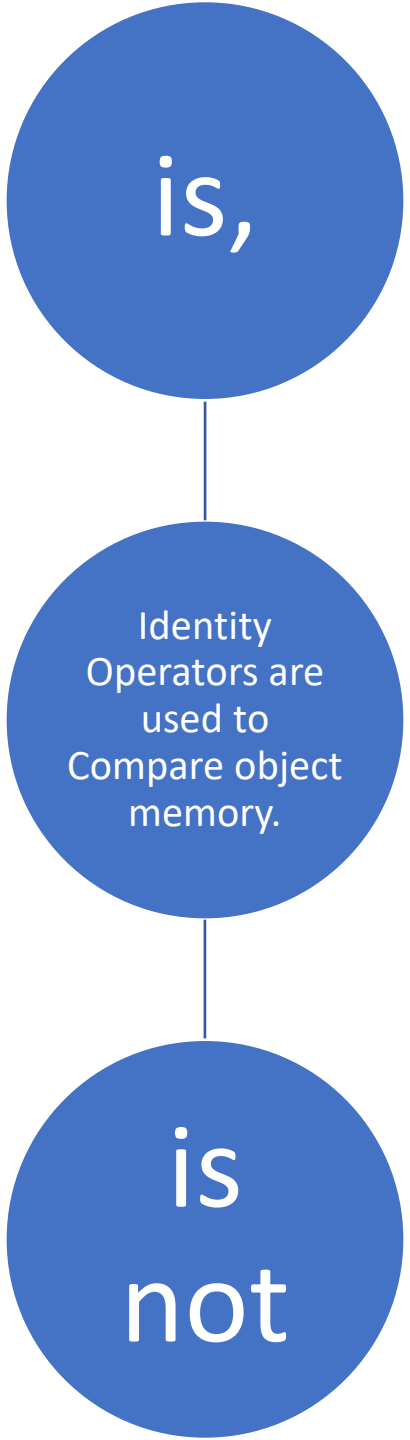
Bitwise Operators


&,
<<
|,
Bitwise Operators are used to Manipulate binary data.
>>,
~,
^,

Membership Operators


in,
Membership Operators are used to Check if in a sequence.
not in

Identiy Operators


is,
Identity Operators are used to Compare object memory.
is not

int + int=int

str + str=str

float + float=float

int + float=float

complex + complex=complex

bool + Boolean=int

tuple + tuple=tuple

list + list=list

complex + int=complex

complex + bool=complex

float + boolean=float

Addition operator doesn't support for set dictionary range.

Addition operator doesn't work for complex string--------->complex + string.

float + list
float + set
float + dictionary
float + tuple
int + string
int + list
int + dict.
int + tuple
string + different datatypes
list + different datatypes
tuple + different datatypes

Subtraction(-)

For Same datatypes.

For Strings Subtraction operation doesn’t work for string.

For Same datatypes:

list - list

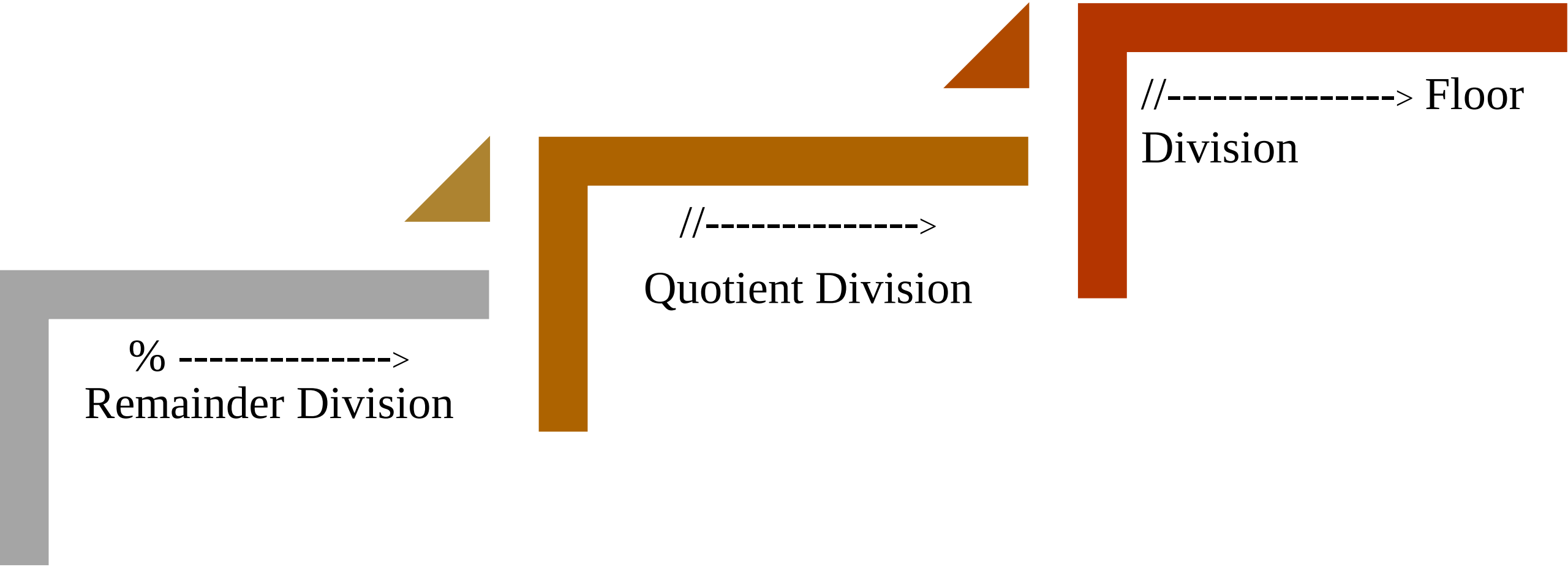
string - string

array - array

set - set

dictionary - dictionary

tuple - tuple


% ----------> Remainder Division
//----------> Quotient Division
//----------> Floor Division

Precedence and Associativity

PEMDAS

- P Parenthesis ()
- E Exponent **
- M Multiplication *
- D Division /
- A Addition +
- S Subtraction –

BODMAS

- B Brackets [ ]
- O Order of powers
- D Division /
- M Multiplication *
- A Addition +
- S Subtraction-

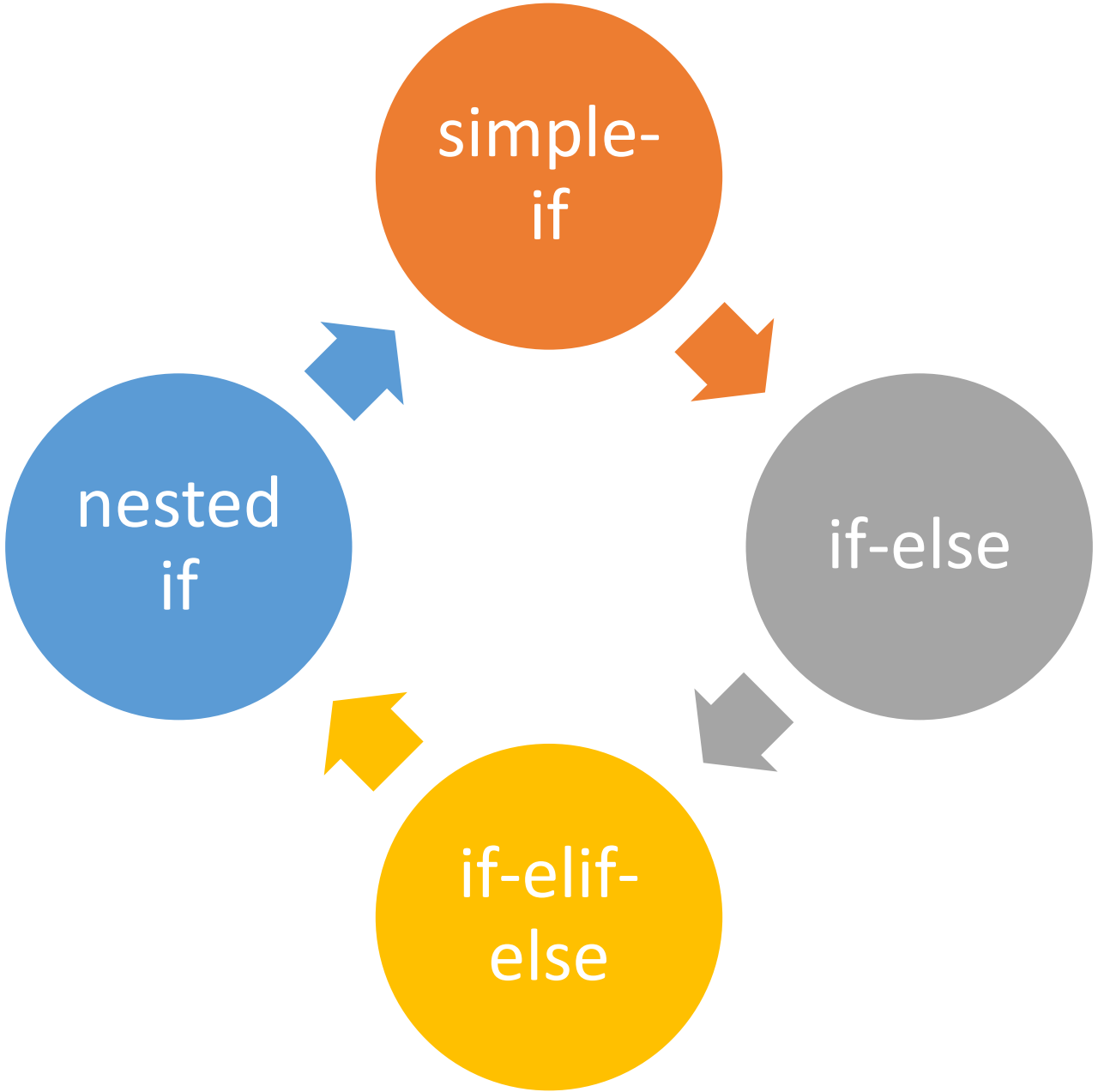
S.NO	OPERATOR	OPERATOR NAME	ASSOCIATIVITY
1	( )	paranthesis	Left to Right
2	**	Exponent	Right to Left
3	*, /, %	Multiplication , division , modulo division	Left to Right
4	+ , -	Addition , subtraction	Left to Right
5	<< , >>	Bitwise Left shift , Bitwise Right Shift	Left to Right

S.NO	OPERATOR	OPERATOR NAME	ASSOCIATIVITY
6	< , <= , > , >=	Less than , Less than or equal to , greater than , greater than or equal to	Left to Right
7	==, !=	is equal to , is not equal to	Left to Right
8	is , is not, in , not in	Identity opertors Membership operators	Left to Right
9	&	Bitwise AND	Left to Right
10	^	Bitwise Exclusive OR	Left to Right

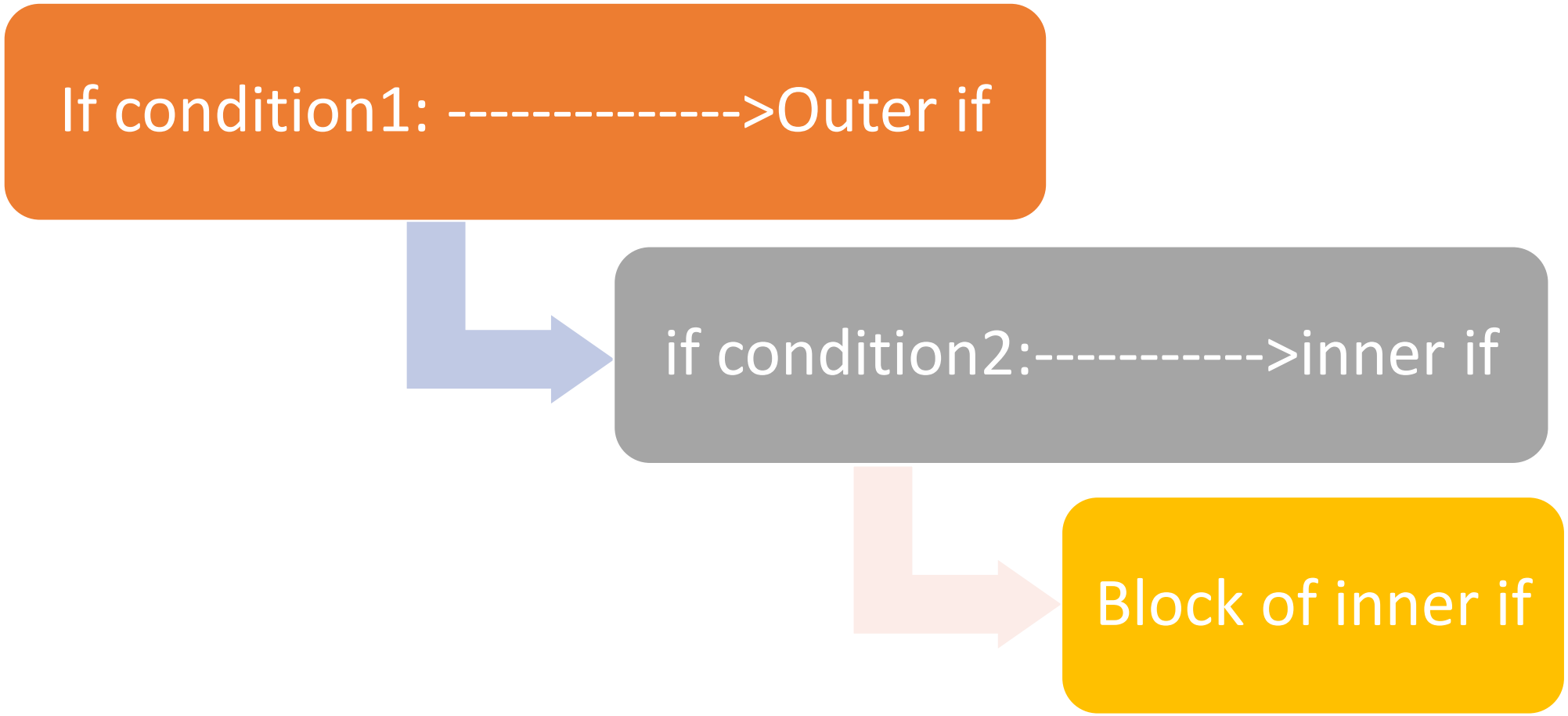
S.NO	OPERATOR	OPERATOR NAME	ASSOCIATIVITY
11	|	Bitwise OR	Left to Right
12	not	Logical NOT	Right to Left
13	and	Logical AND	Left to Right
14	or	Logical OR	Left to Right

S.NO	OPERATOR	OPERATOR NAME	ASSOCIATIVITY
1	+=, -=, *=, /=, %=, &=, ^=, |=, <<=, >>=	Addition assignment, subtraction assignment, multiplication assignment, division assignment, modulo division assignment, bitwise AND assignment, bitwise Exclusive or assignment Bitwise or assignment, bitwise left shift assignment, bitwise right shift assignment.	Left to Right

List of types of conditional statements:

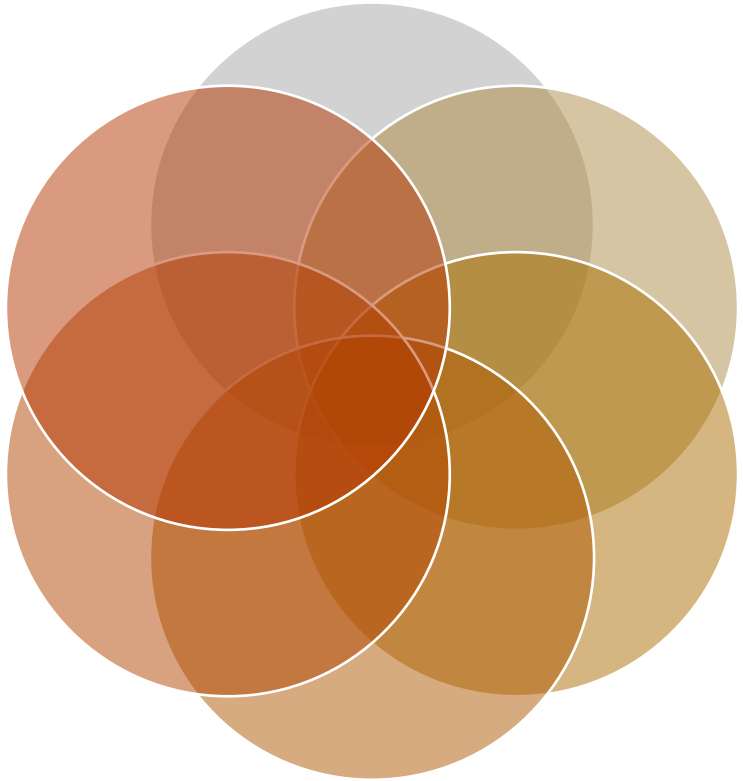


Syntax for Nested - if :



iterative statements

or

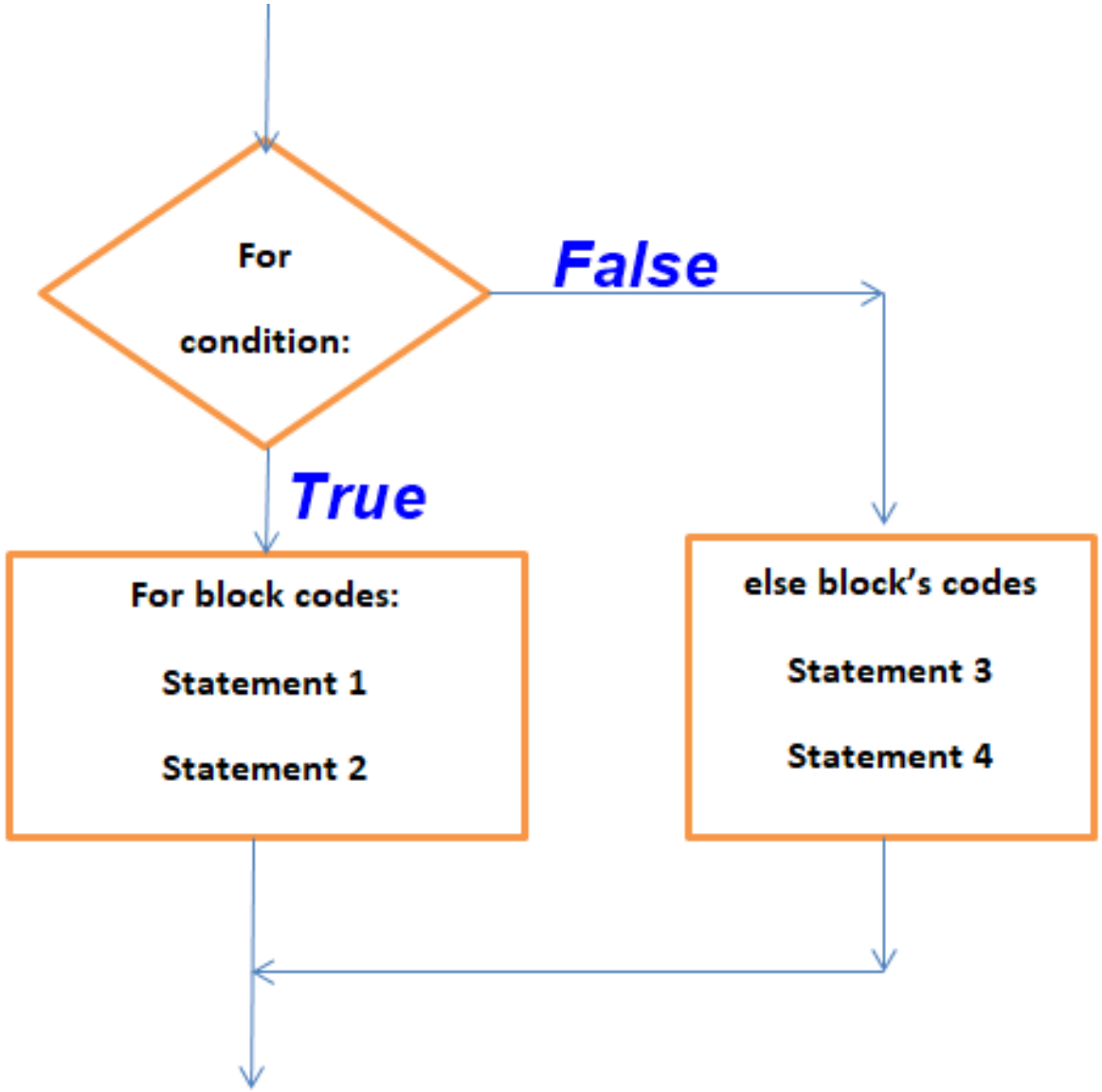


Repetitive statements

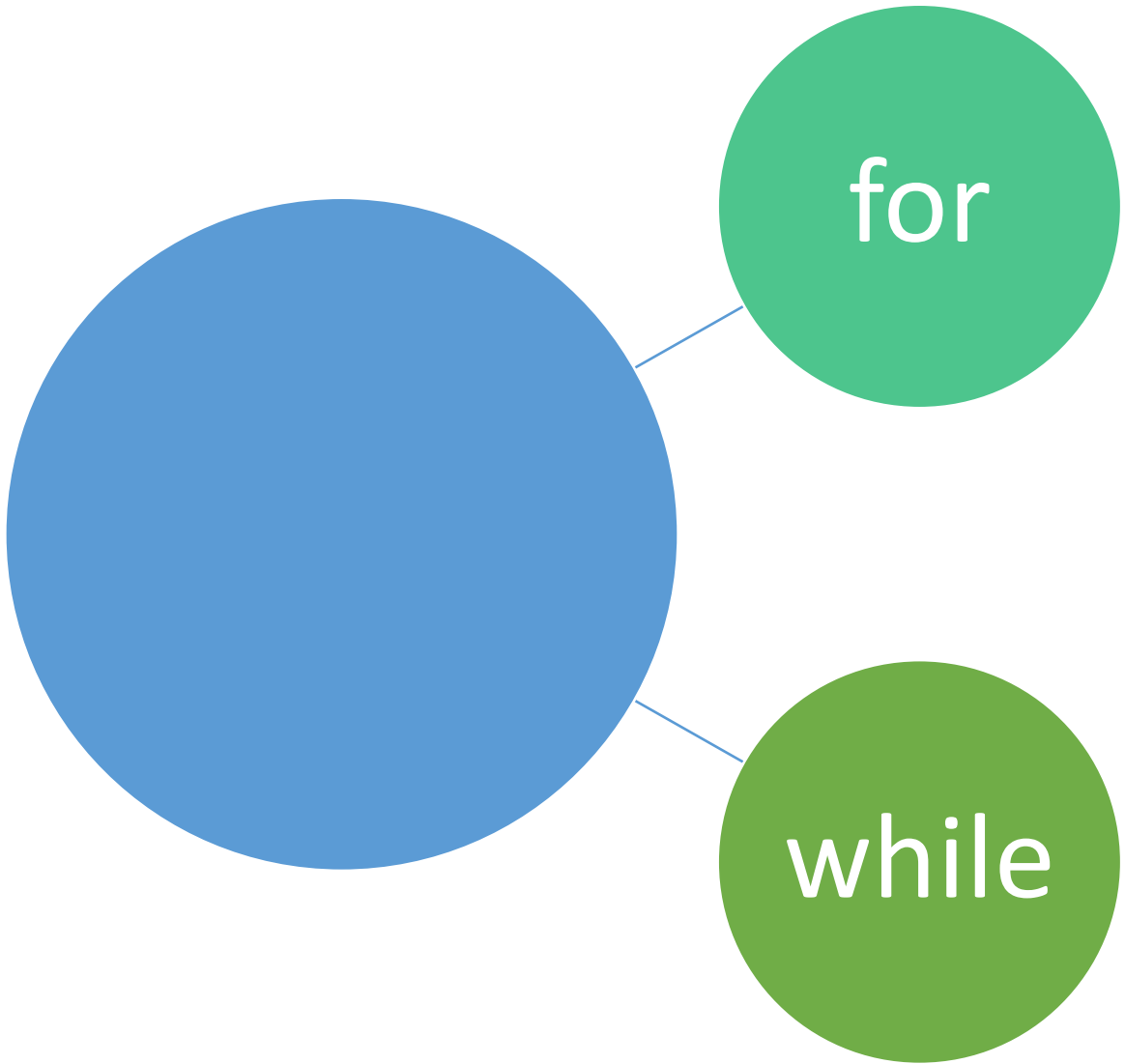
or

loop statements

Repetitive Statement:



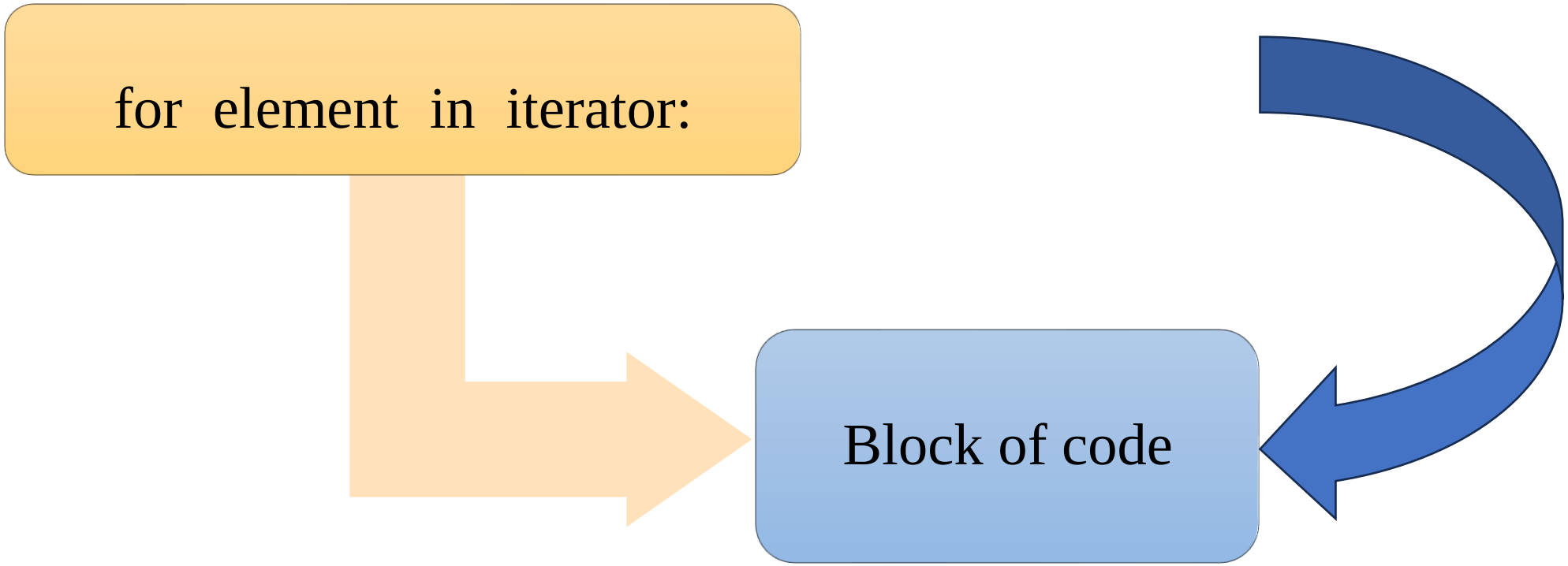
To create loop statements we want two keywords

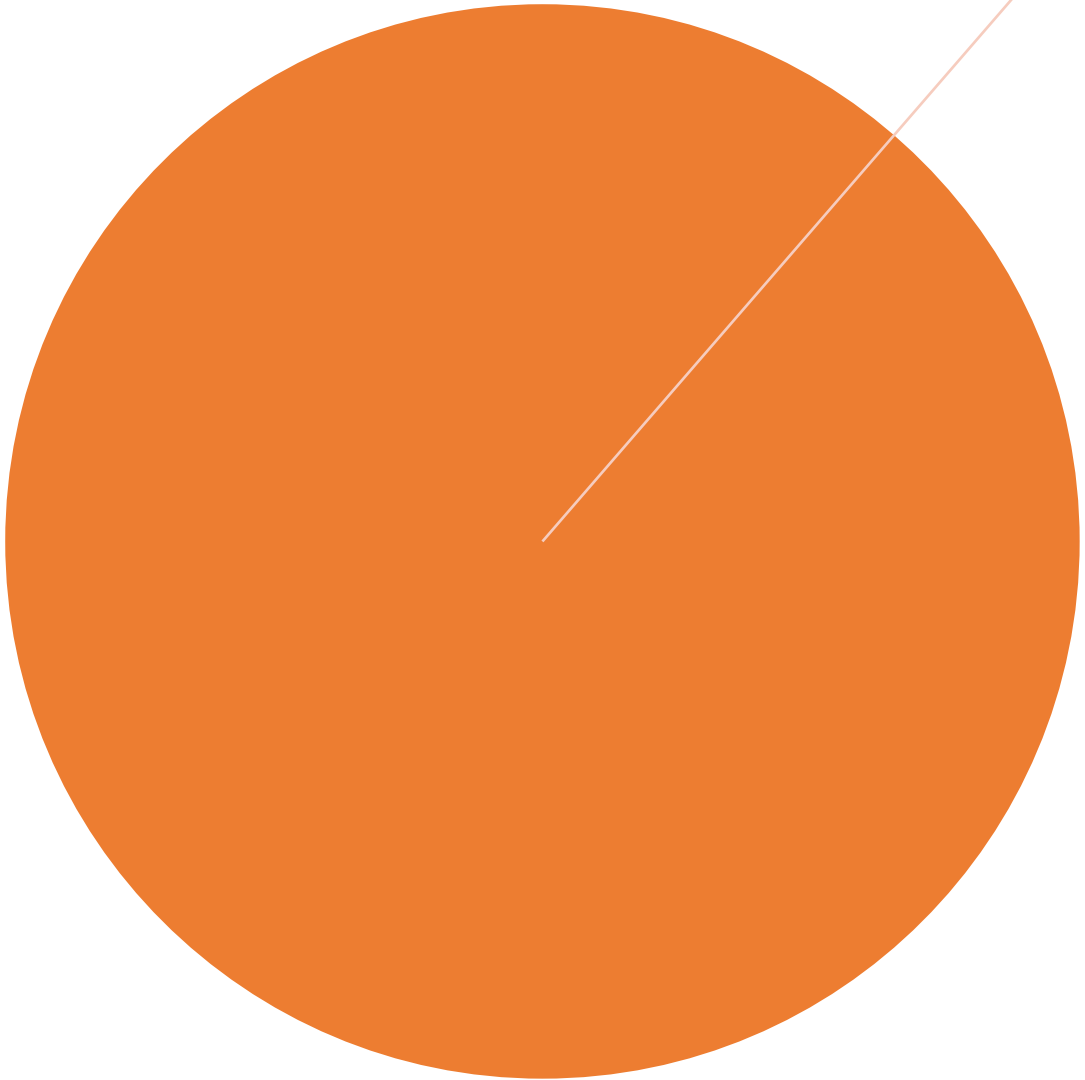


for loop

- for loop is used to iterate the iterator or range.
- iterators - list set tuple dictionary array string etc…
- range is object range ( )

Syntax of for loop:

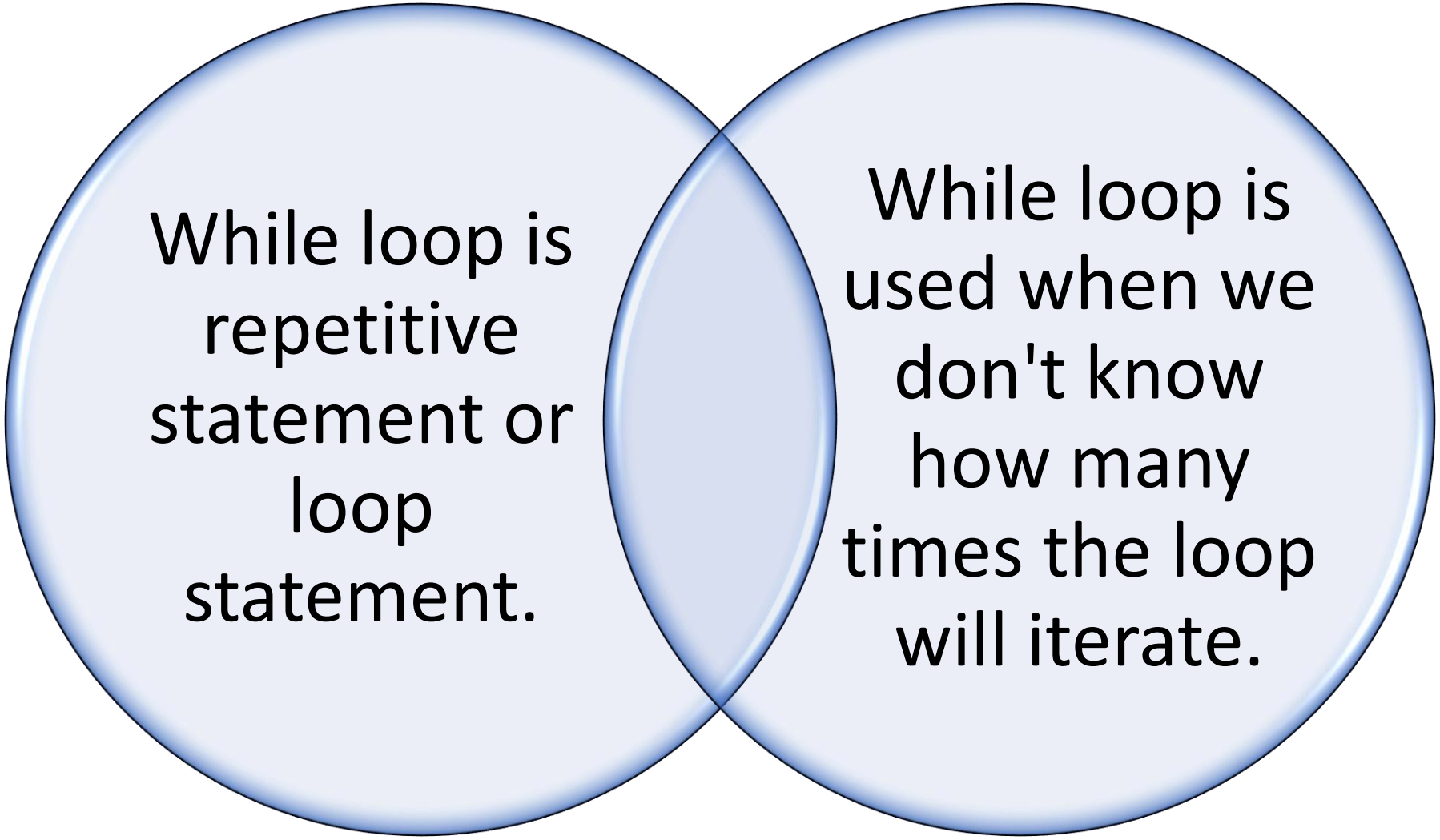




for loop is used to iterate the block of code in given number of times.

While loop

While loop


While loop is repetitive statement or loop statement.
While loop is used when we don't know how many times the loop will iterate.

Syntax of while loop:

```
initialization
while condition:
    print(i)
    increment
print('loop completed)
```

Data Structures

Built_in

List

Tuple

Set

Dictionary

User defined

Stack

Queue

Linked list

Tree

Graph