PYTHON BASICS

What is Python?

- Python is a General-Purpose Programming language that is often applied in scripting roles.
- So, Python is programming language as well as scripting language.
- Python is an Interpreted language

Why was Python Created?

"My original motivation for creating Python was the perceived need for a Higher-level language in the Amoeba [Operating Systems] project.

I realized that the development of system administration utilities in C was taking too long. Moreover, doing these things in the Bourne shell wouldn't work for a variety of reasons.

So, there was a need for a language that would bridge the gap between C and the shell"

- Guido Van Rossum

Python - Uses

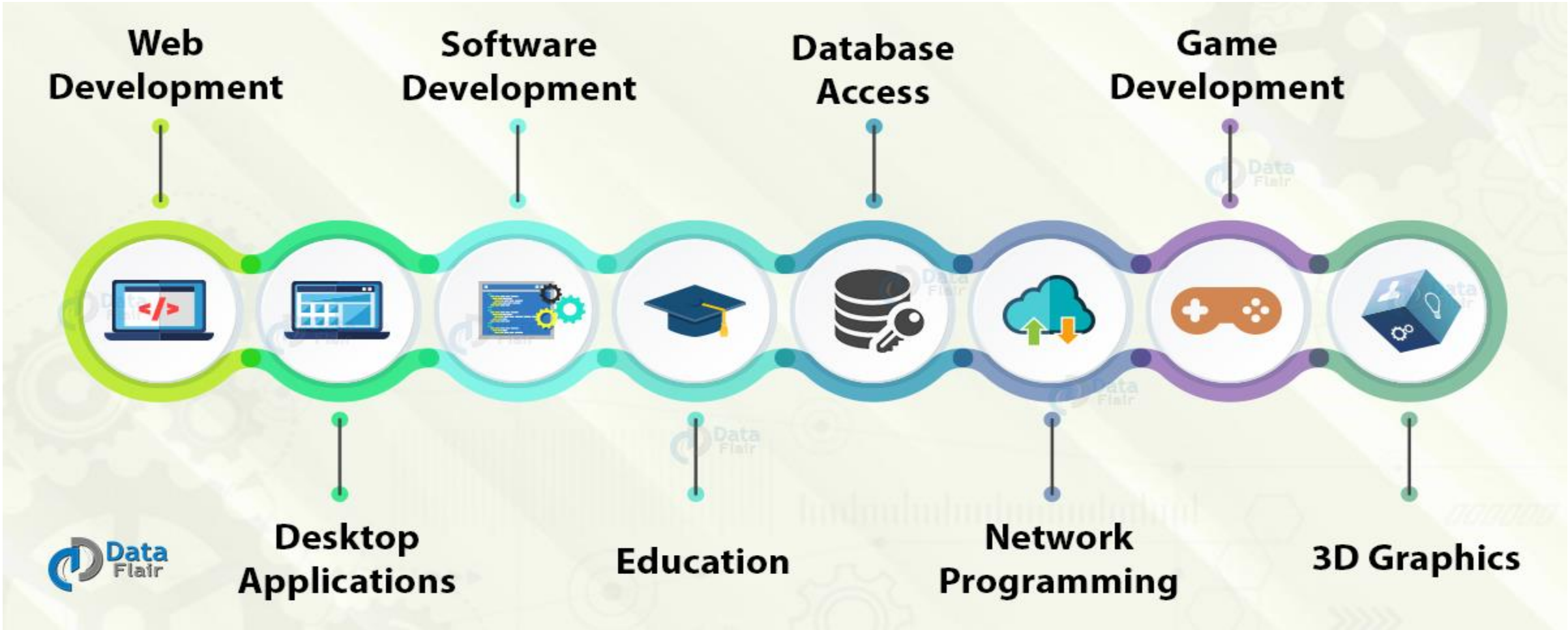
WEB DEVELOPMENT

SOFTWARE DEVELOPMENT

MATHEMATICS

DATA SCIENCE

Python - Application



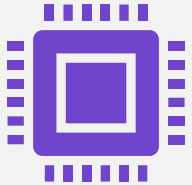
Hands On Activities for Python

- Variables
- Datatypes(Number, Boolean, String, Tuple)
- Datatypes(Set and List)
- Condition and Loop
- Functions and Methods

Variables

Variables are containers for storing data values

Variables and data types in python as the name suggests are the values that vary



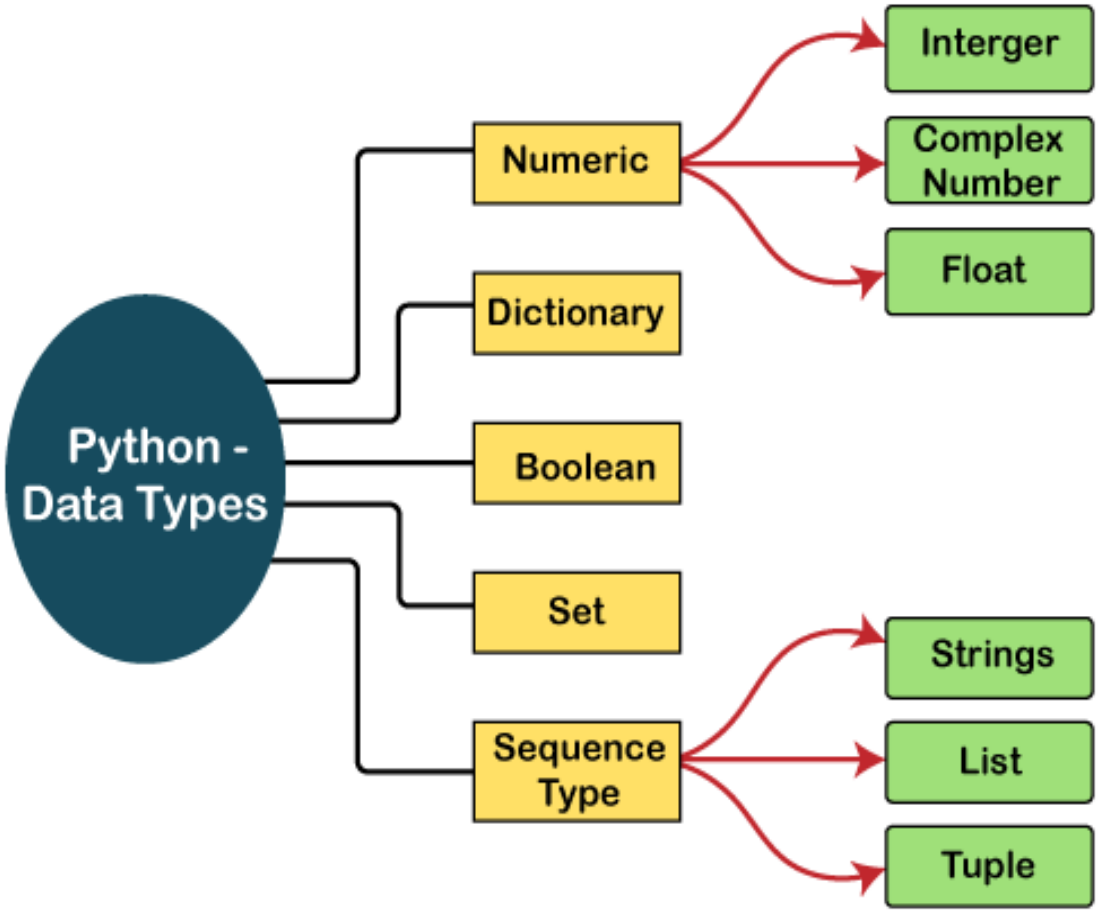
A Variable in python is created as soon as a value is assigned to it

What are the Variables in python?

- A Python variable is a reserved memory location to store values.
- This means that when you create a variable you reserve some space in memory.
- Every value in Python has a datatype.
- We do not need to declare variables before using them or declare their type.
- Variables can be declared by any name or even alphabets like a, aa, abc, etc.

What are the Datatypes in Python?

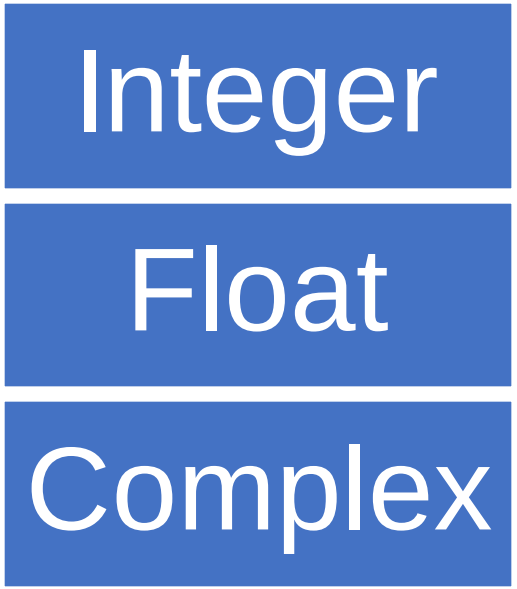
- Numeric
- Sequence Type
- Boolean
- Set
- Dictionary



http://www.javatpoint.com/python-data-types

Numeric

- In Python, numeric data type represent the data which has numeric value. Numeric value can be integer, floating number or even complex numbers. These values are defined as int, float and complex class in Python.



Integers

- There are two types of integers in Python:

- Integers(Signed) : It is the normal integer representation of whole numbers using the digits 0 to 9. Python provides single int data type to store any integer whether big or small. It is signed representation i.e. it can be positive or negative.

- Boolean : These represent the truth values True and False. It is a subtype of integers and Boolean values True and False corresponds to values 1 and 0 respectively

Numeric

- **Float** – This value is represented by float class. It is a real number with floating point representation. It is specified by a decimal point. Optionally, the character e or E followed by a positive or negative integer may be appended to specify scientific notation.

- **Complex Numbers** – Complex number is represented by complex class. It is specified as *(real part)* + *(imaginary part)j*. For example – 2+3j

Datatypes in Python

a = Assign value (Int/float/complex)

print("Type of a: ", type(a))

Integers – represented by int class

Float – represented by float class

Complex Numbers – represented by complex class

Numeric

Coding

```
a = 5
print("Type of a: ", type(a))

b = 5.0
print("\nType of b: ", type(b))

c = 2 + 4j
print("\nType of c: ", type(c))
```

Output

```
Type of a:  <class 'int'>

Type of b:  <class 'float'>

Type of c:  <class 'complex'>
```

Sequence Type

- String
- List
- Tuple

String

```
String1 = "Welcome to Python's World"
print("\nString with the use of Double Quotes: ")
print(String1)
print(type(String1))
```

Arrays of bytes representing Unicode characters

Collection of one or more characters put in a single, double or triple quote

No character data type, a character is a string of length one

List

- Lists are just like the arrays, declared in other languages which is an ordered collection of data
- It is very flexible as the items in a list do not need to be of the same type

Creating List

Lists in Python can be created by just placing the sequence inside the square brackets

```
List = [['Welcome', 'To'], ['Python']]
print("\nMulti-Dimensional List: ")
print(List)
```

Functions in Python

- A block of organized, reusable code used to perform a single, related action
- Python gives us many built-in functions like print , etc. but you can also create your own functions called user-defined functions

Syntax def my_function():

Tuple

Just like list, tuple is also an ordered collection of Python objects

Tuples are immutable i.e. tuples cannot be modified after it is created

It is represented by tuple class

Creating Tuple

Note: Tuples can also be created with a single element, but it is a bit tricky

```
Tuple1 = ()
print("Initial empty Tuple: ")
print (Tuple1)
```

Boolean

- Returns True or False

```
x = bool(5)
print(x)
 print(type(x))
```

Set

- Set is an unordered collection of data type that is iterable, mutable and has no duplicate elements
- Sets can be created by using the built-in set function’

```
myset = {"apple", "banana", "cherry"}
print(myset)
```

Dictionary

- Dictionary in Python is an unordered collection of data values, used to store data values like a map
- Dictionary holds key:value pair
- Key-value is provided in the dictionary to make it more optimized
- Each key-value pair in a Dictionary is separated by a colon :, whereas each key is separated by a ‘comma’

Creating Dictionary

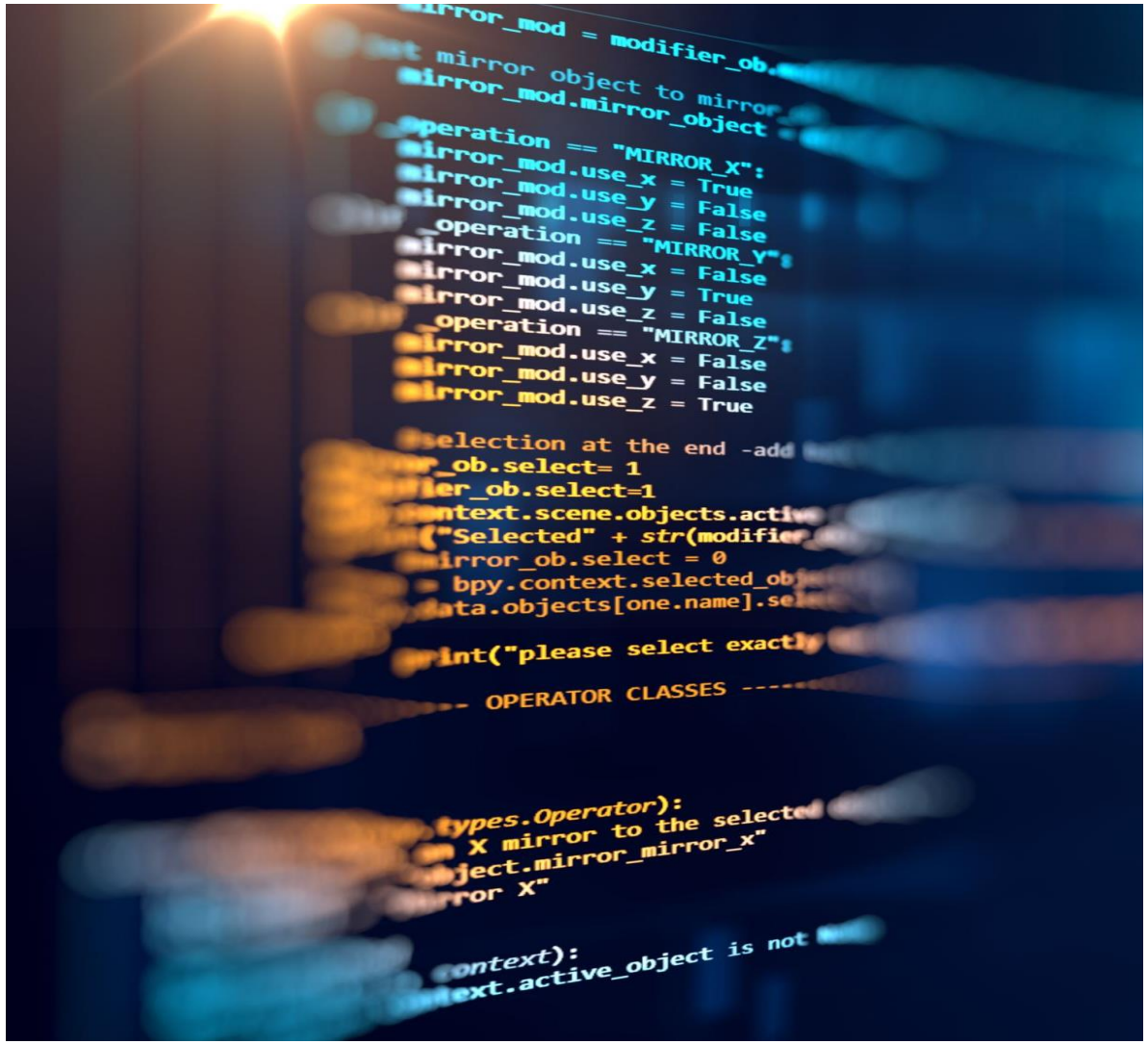
- Note – Dictionary keys are case sensitive, same name but different cases of Key will be treated distinctly

```
Dict = {}
print("Empty Dictionary: ")
print(Dict)
```

Strings & Methods

- Capitalize
- Casefold
- Center
- Count
- Encode
- Endswith
- Upper

Microsoft SAP edunet foundation



Format

- Handle complex string formatting

```
txt = "For only {price:.2f} dollars!"
print(txt.format(price = 49))
```

Relational Operators

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

If Condition

- Used for Decision making.
- If you have only one statement to execute, you can put it on the same line as the if statement
- Remember to use Indentation properly and semicolon.

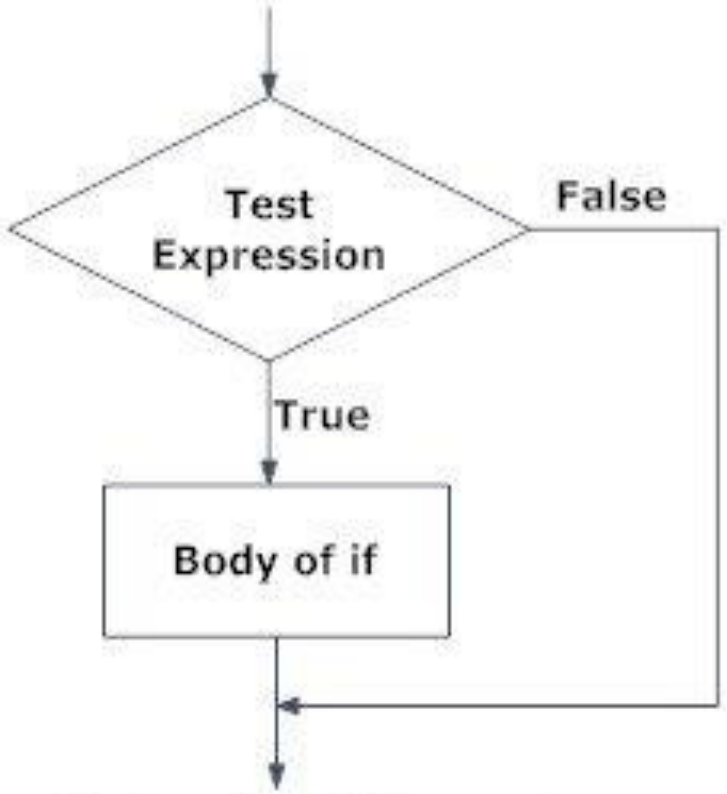


Fig: Operation of if statement

Else

The else keyword catches anything which isn't caught by the preceding conditions

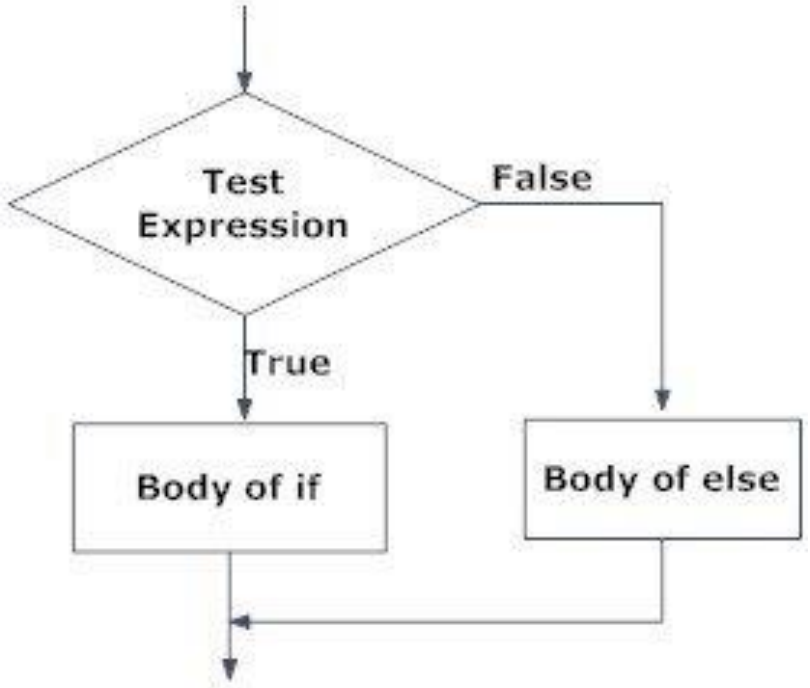


Fig: Operation of if...else statement

Elif

The elif keyword is python's way of saying "if the previous conditions were not true, then try this condition"

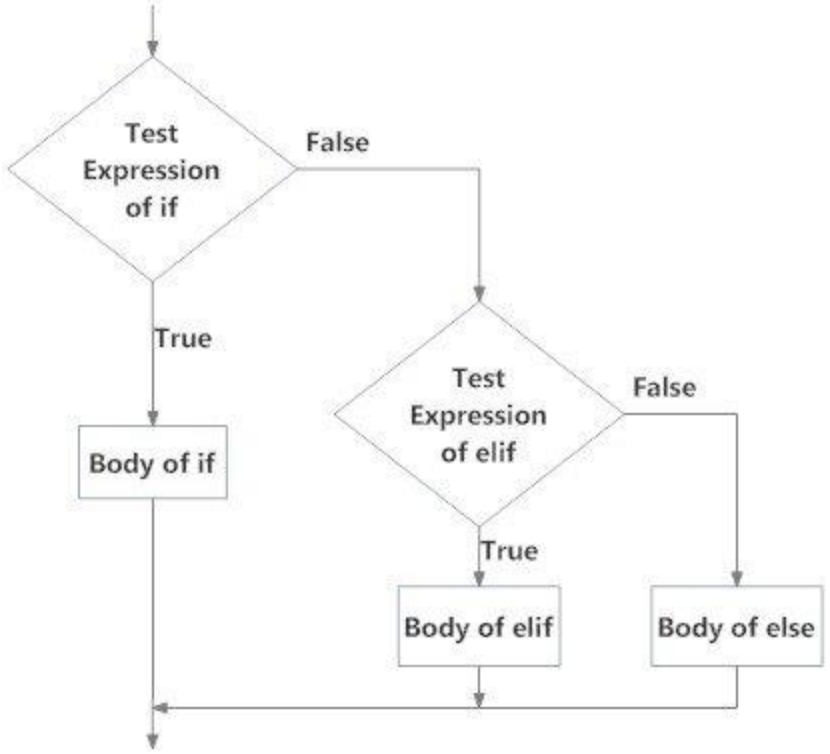


Fig: Operation of if...elif...else statement

If, Elif & Else Statement

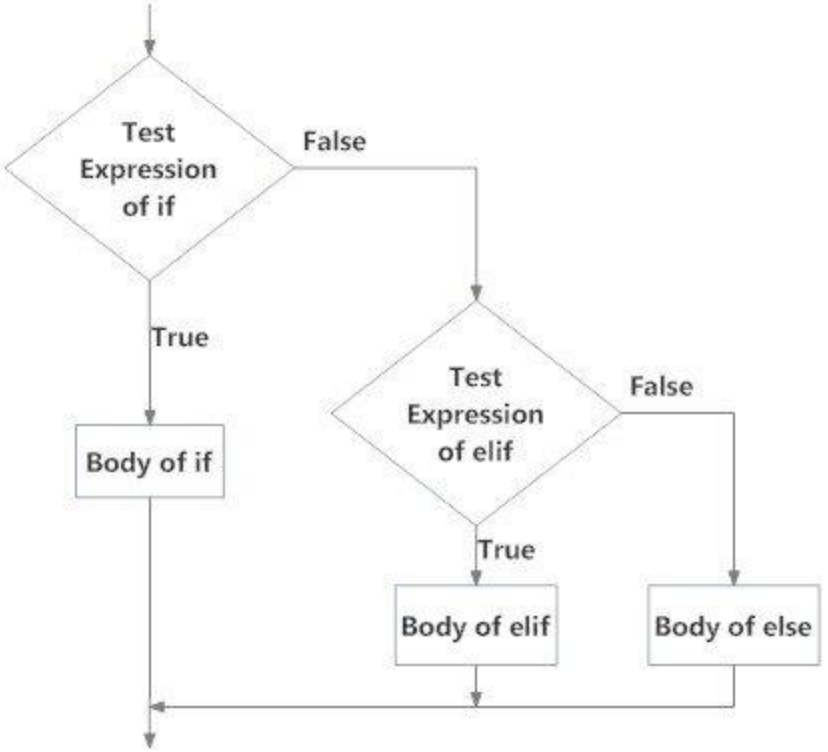


Fig: Operation of if...elif...else statement

Range Function and for loop

- To loop through a set of code a specified number of times, we can use the range function
- The range function returns a sequence of numbers, starting from 0 by default, and increments by 1 , and ends at a specified number

```
for x in range(6):
  print(x)
```

Nested Loops

- A nested loop is a loop inside a loop
- The "inner loop" will be executed one time for each iteration of the "outer loop“

```
adj = ["red", "big", "tasty"]
fruits = ["apple", "banana", "cherry"]

for x in adj:
  for y in fruits:
    print(x, y)
```

Break Statement

- With the break statement we can stop the loop before it has looped through all the items

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
 print(x)
if x == "banana":
  break
```

Continue and Pass Statement

- Stop the current iteration of the loop, and continue with the next

```
fruits = ["apple", "banana", "cherry"]
        for x in fruits:
          if x == "banana":
            continue
print(x)
```

- For a blank loop

```
for x in [0, 1, 2]:
  pass
```

Python Libraries

- Python Libraries are a set of useful functions that eliminate the need for writing codes from scratch
- There are over 137,000 python libraries present today
- Python libraries play a vital role in developing machine learning, data science, data visualization, image and data manipulation applications and more

What is a Library?

- A library is a collection of pre-combined codes that can be used iteratively to reduce the time required to code
- They are particularly useful for accessing the pre-written frequently used codes, instead of writing them from scratch every single time
- Similar to the physical libraries, these are a collection of reusable resources, which means every library has a root source
- This is the foundation behind the numerous open-source libraries available in Python

NumPy

- Supports large multidimensional arrays and matrices
- High-level mathematical functions
- NumPy relies on BLAS and LAPACK for efficient linear algebra computations
- NumPy can also be used as an efficient multi-dimensional container of generic data

Pandas

It is an open-source, BSD licensed library

Pandas enable the provision of easy data structure and quicker data analysis for Python

For data analysis and modelling, Pandas have become the first choice

Scikit- learn

- A free software machine learning library for the Python programming language
- Can be effectively used for a variety of applications which include classification, regression, clustering, model selection, naive Bayes', grade boosting, K-means, and pre-processing

TensorFlow

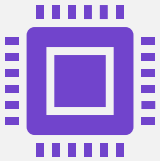
- The most popular deep learning framework
- Open-source software library for high-performance numerical computation
- Math library used for machine learning and deep learning algorithms
- Tensorflow was developed by the researchers at the Google Brain team within Google AI organization
- Used by researchers for machine learning algorithms, and by physicists for complex mathematical computations

Keras

- Open-source neural network library for fast experimentation with deep neural networks
- With deep learning becoming ubiquitous, Keras becomes the ideal choice as it is API designed for humans and not machines according to the creators

OpenCV Python

Open Source Computer Vision or OpenCV is used for image processing



Monitors overall functions focused on instant computer vision

Provides several inbuilt functions, to learn/develop Computer Vision

Installing Packages with pip

Check, whether you have an installed version

py –version

If you need help, please try help

Install the required packages

Attributes

- Class attributes belong to the class itself they will be shared by all the instances
- Such attributes are defined in the class body parts usually at the top, for legibility

Instance Attributes

Unlike class attributes, instance attributes are not shared by objects

Every object has its own copy of the instance attribute

It also displays the attributes of its ancestor classes

REFERENCES

1. https://en.wikipedia.org/wiki/Anaconda_(Python_distribution)
2. https://docs.python.org/3/library/
3. https://www.tutorialspoint.com/numpy
4. https://numpy.org/
5. https://towardsdatascience.com/
6. https://realpython.com/

Microsoft

SAP

edunet foundation

THANK YOU