

PYTHON- BASICS

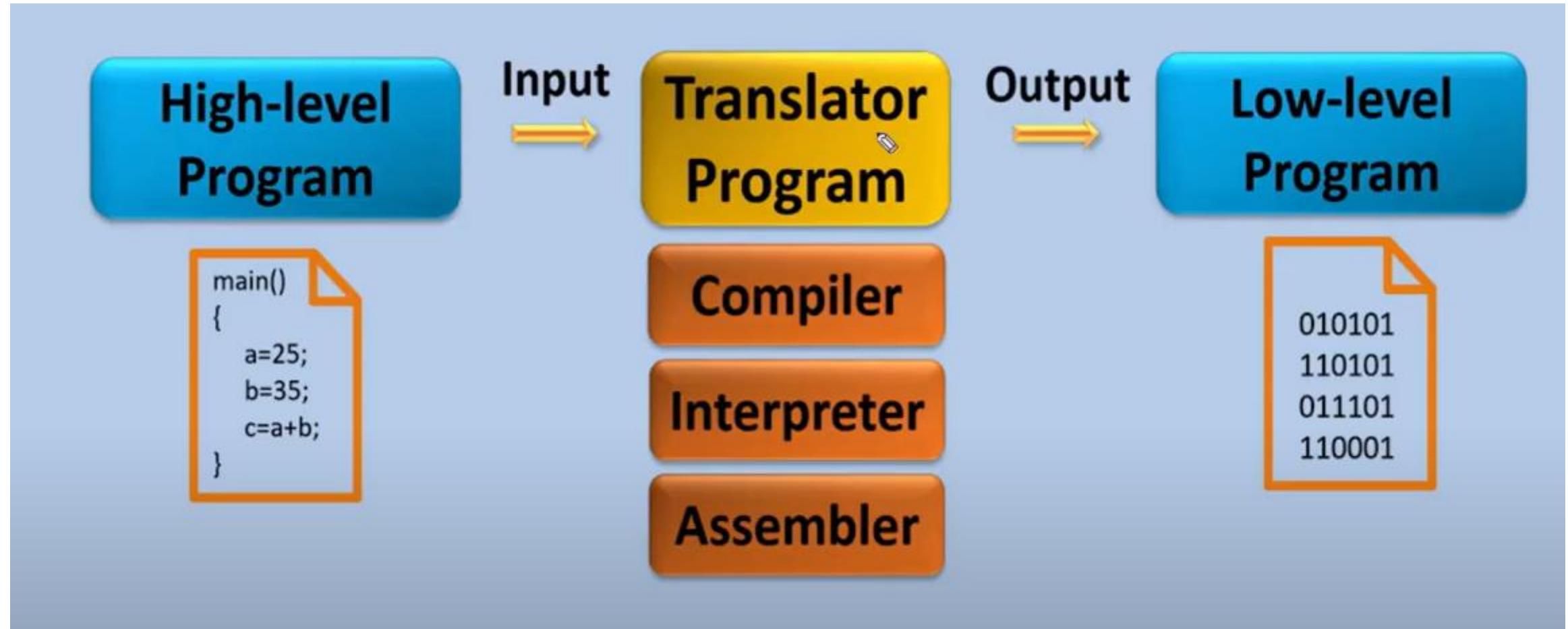
Dr Sivabalan,
Technical Training Advisor
Sivabalan.n@nttdata.com

programming languages:-

Each programming language has its own syntax and rules

1. Low-Level Languages
2. High-Level Languages

Python-Basics



What is Python?

Python is a high level programming language like C,C++

Python is a simple, high level, interpreted, general purpose, dynamically typed and object oriented programming language created by Guido Van Rossum in 1991.

Python is simple

- Python is easy to use.
- python's programs are simple to write and hence it is easy to learn.
- Lots of built-in modules, packages and Frameworks

Python-Basics

Hello World

Java:

```
// Hello World in Java
class HelloWorld {
    static public void main(String args[]) {
        System.out.println("Hello World!");
    }
}
```

C++:

```
// Hello World in C++
#include <iostream.h>
Main() {
    cout << "Hello World!" << endl;
    return 0;
}
```

Python:

```
# Hello World in Python
print("Hello World!")
```

Python combines remarkable power with very clean, simple, and compact syntax.

Python is high-level

- More user-friendly
- There is automatic memory management
- Rich set of libraries and functions ex. len()

Python is dynamically typed

- Like C or C++, we don't need to specify datatypes of identifiers.
- Python evaluates datatypes at runtime

Python is general purpose programming language:-

Python is used for web development, machine learning, artificial intelligence, data analysis, data science, web scraping, scripting, scientific computing, software dev etc

Applications of Python

Software Applications

Web Development

Data Science

Data Analysis

Machine Learning

Artificial Intelligence

Web scraping

Image Processing

Game Development

Scripting

Network Programming

Scientific & Numeric

Console & GUI based apps

Python-Basics

Web applications :- Django, flask, pyramid Frameworks

GUI application:- libraries like Kivy, Tkinter

Scientific and numeric applications:- Scipy, pandas, numpy

Data science:- numpy, pandas, seaborn, matplotlib

Image processing:- OpenCv library

Game Development:- Pygame library

- What are variables and need of variables ?
- How to create variables & working of variables ?
- Rules for creating variables (Naming conventions)
- do's & dont's
- Keywords in python

How variable works in C ?

```
int a = 20;    int b = 20;    int c = a;    int d = 30;
```

a

20

1244

b

20

1248

c

20

1252

d

30

1256

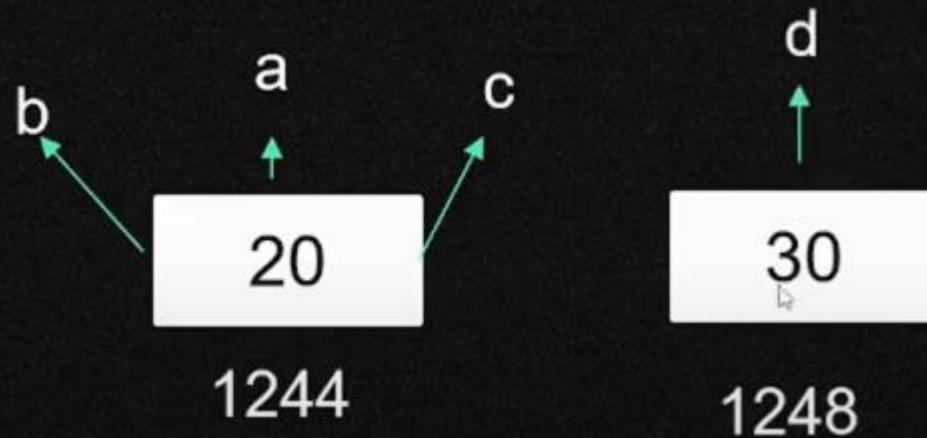
How variable works in python ?

a = 20;

b = 20;

c = a;

d = 30;



How variable works in python ?

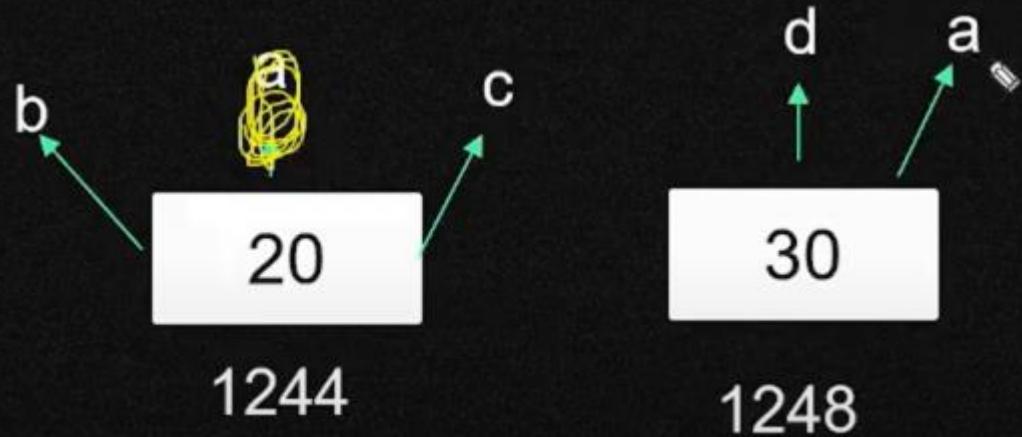
```
a = 20;
```

```
b = 20;
```

```
c = a;
```

```
d = 30;
```

```
a = d;
```



How variable works in python ?

a = 20;

b = 20;

c = a;

d = 30;

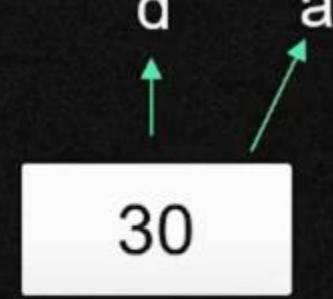
a = d;

b = 40;



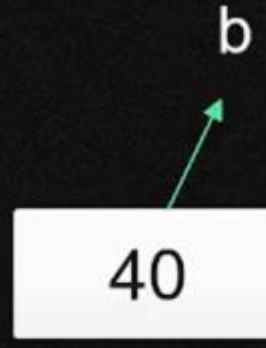
20

1244



30

1248



40

1252

How variable works in python ?

```
a = 20;
```

```
b = 20;
```

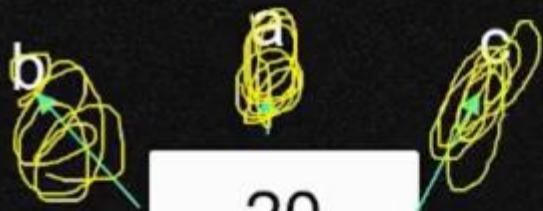
```
c = a;
```

```
d = 30;
```

```
a = d;
```

```
b = 40;
```

```
c = b;
```



How variable works in python ?

```
a = 20;
```

```
a = d;
```



```
b = 20;
```

```
b = 40;
```



```
c = a;
```

```
c = b;
```



```
d = 30;
```

Garbage Collector

Difference between Java/c variables and python variables?

c/java allocates memory for variables.

python allocates memory for values and not for variables.

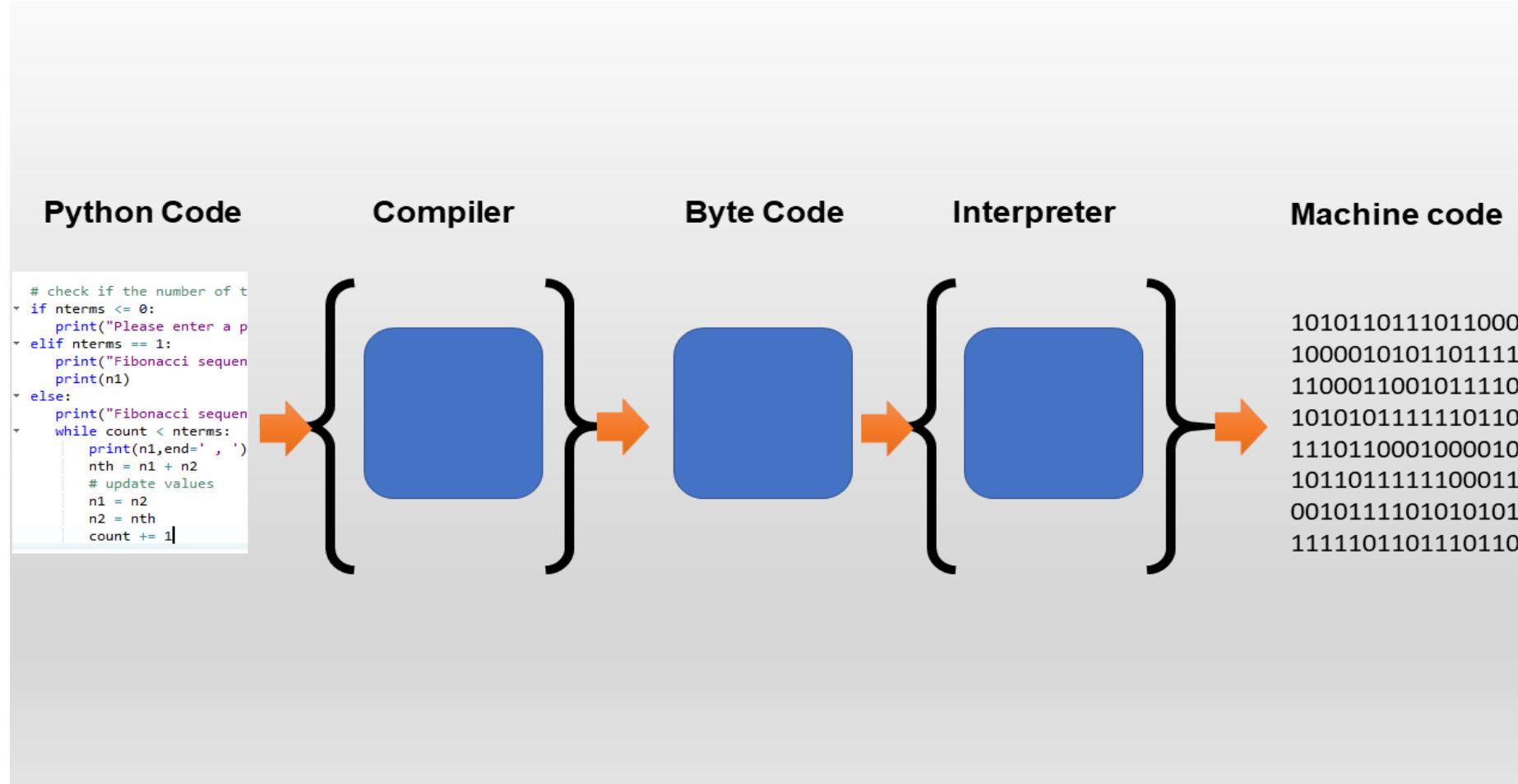
Python has efficient memory management.

Keywords are **special words** in any programming language.

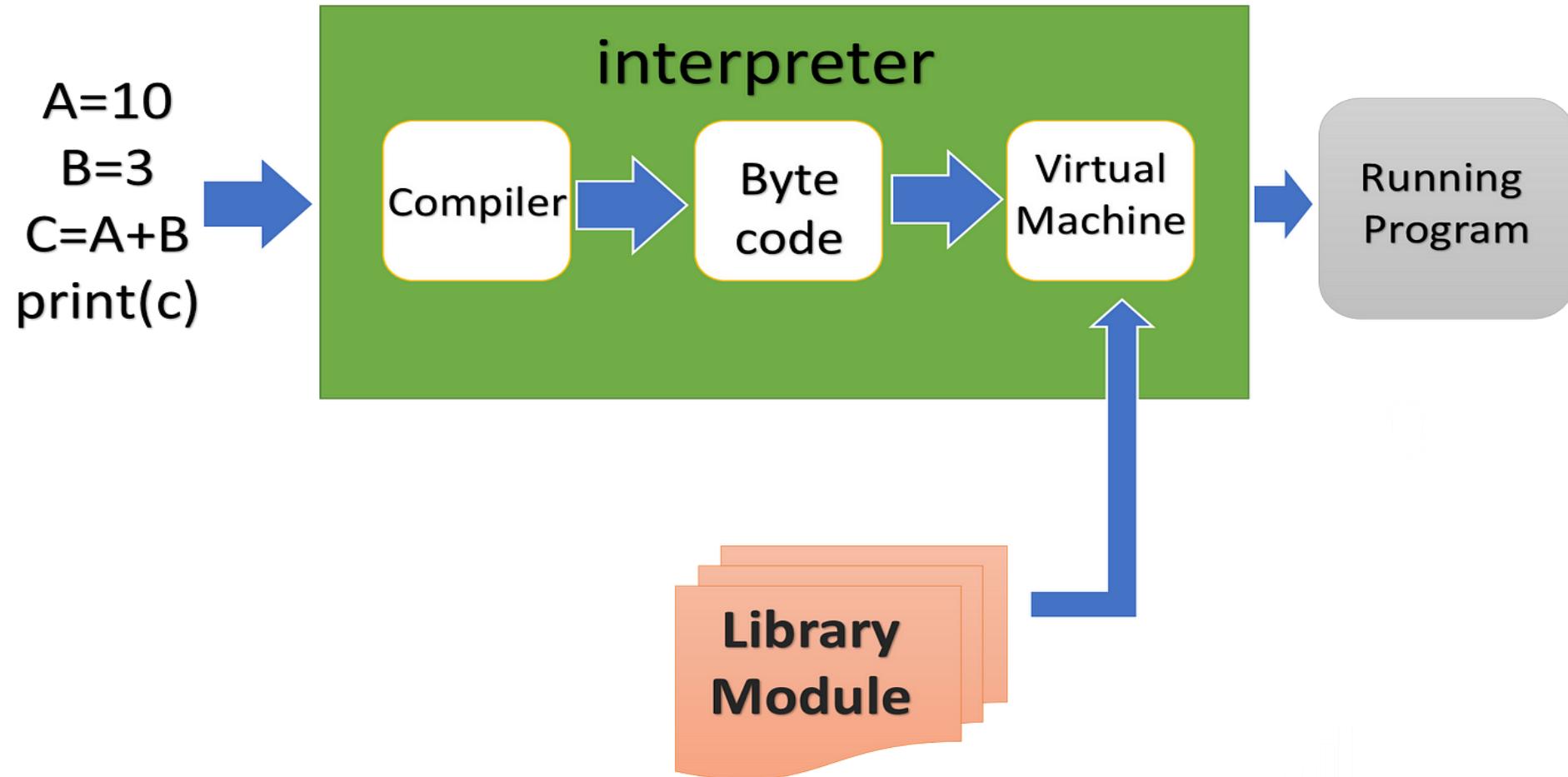
These are **reserved words** in python

Every keyword has specific meaning

Python-Basics



Python-Basics



Comments in Python

An important part of programming

Need of comments

helps you and others to understand later on the intention of your code.

This allows you to more easily find errors, to fix them, to improve the code later on, and to reuse it in other applications as well.

Datatypes in Python

- **type()** function
- **isinstance()** function

Below are datatypes available in python:-

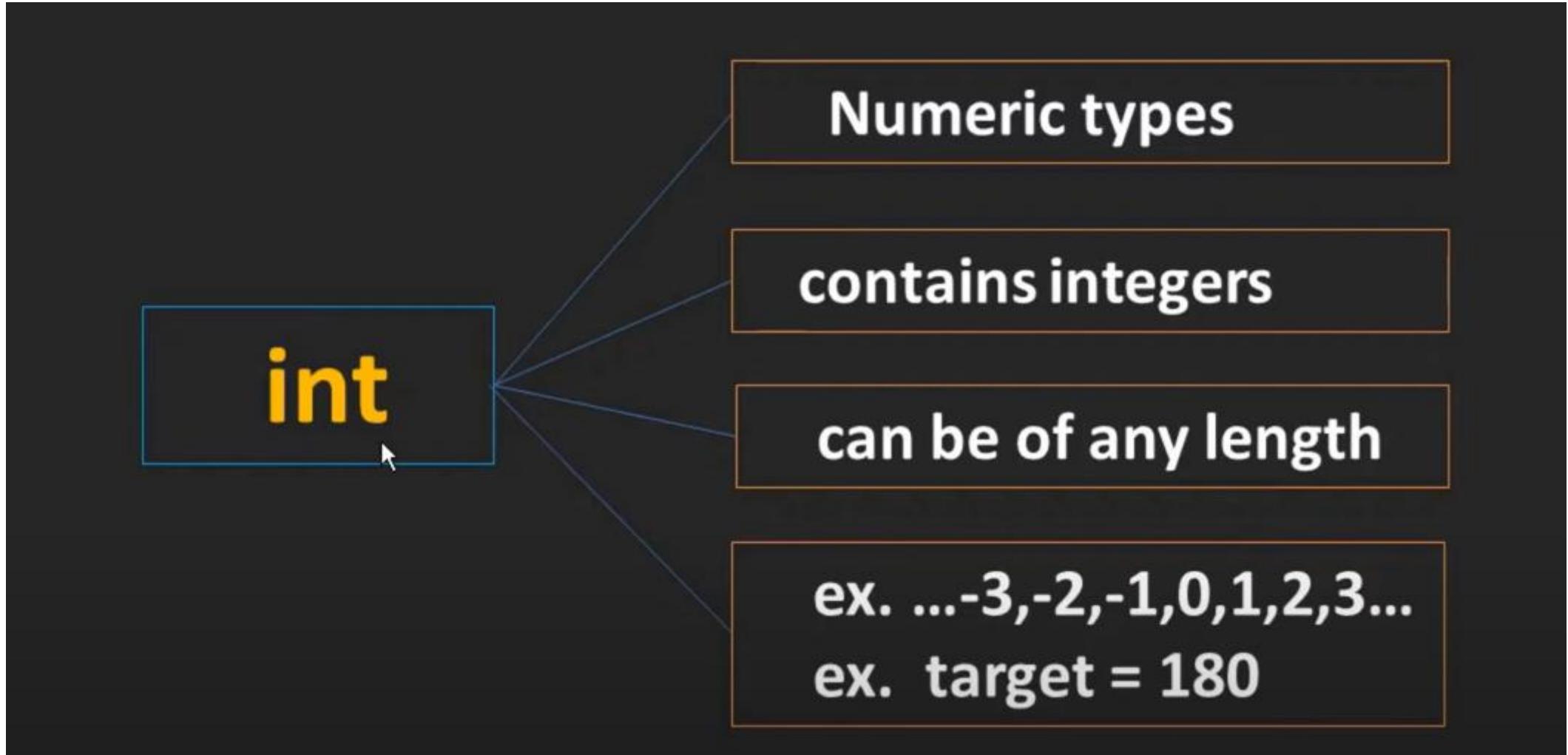
- Numeric types:- int, float, complex (special type)
- Sequence types:- str, list, tuple
- Mapping types:- dict
- Set types:- set, frozenset
- Boolean Type:- Bool
- Binary Types:- bytes, bytearray
- None type:- NoneType

Numeric - types

int

float

complex



float

Numeric type

floating point numbers.

accurate upto 15 decimal places

ex. 3.2, 4.0, 8.9,-12.4

ex. temperature = 37.4

complex

written in form of $a+bj$

‘a’ is real part.can be integer or float.

‘bj’ is an imaginary part.can be integer or float.

Ex. $c = 5+4j$

string

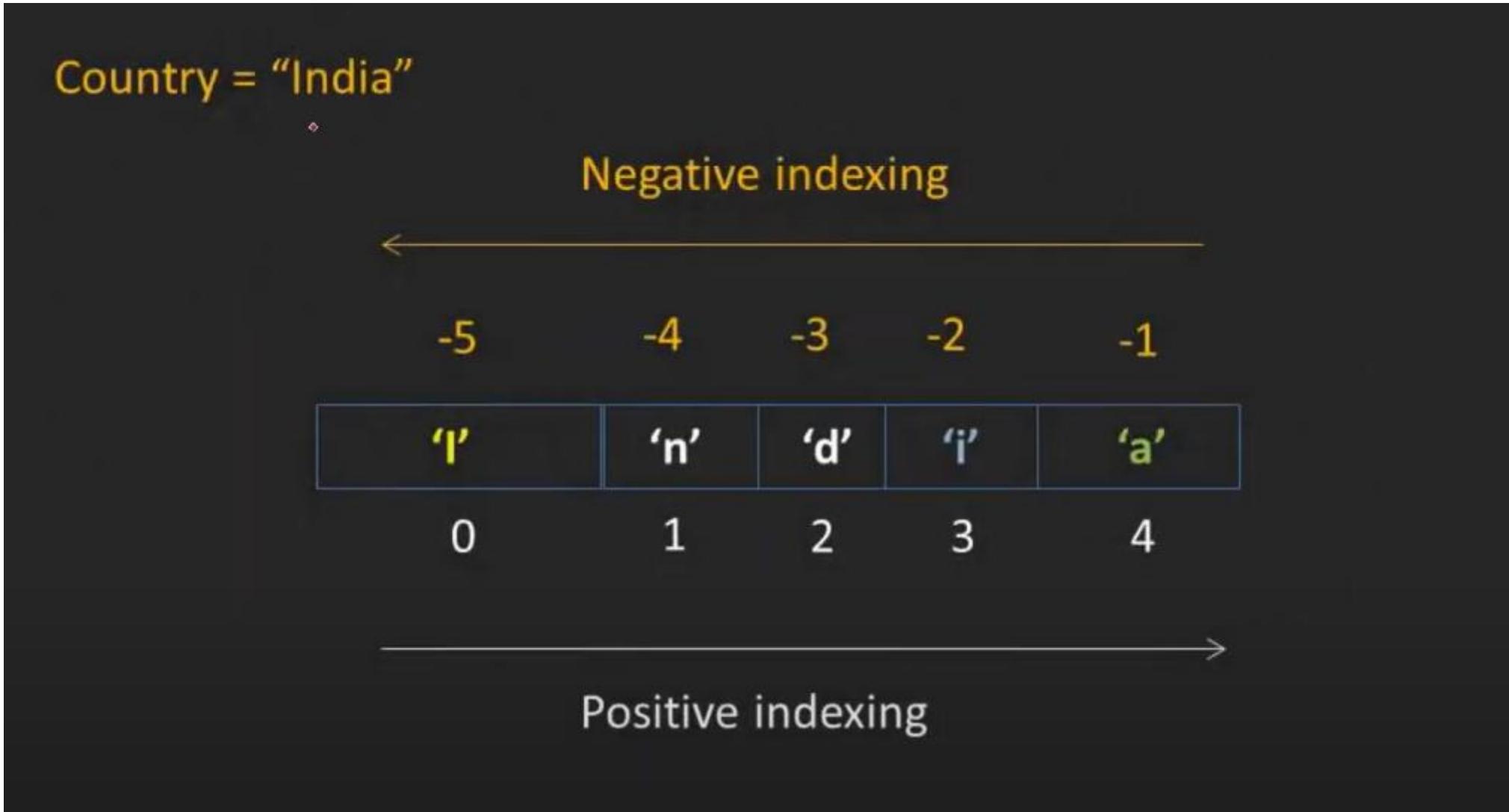
Sequence of characters.

Anything enclosed in quotes.

triple quotes are used for multi-line strings.

Ex. name = “hello”

Python-Basics



sequence - types

list

tuple

dictionary

Set,frozenset

list

list of data's having different data types.

Ordered mutable sequence of items.

Items separated by commas enclosed in [].

Python-Basics

```
My_list = ['facebook', 12.5 ,300, 2+3j , [1,2,3] ]
```

Negative indexing

←

-5 -4 -3 -2 -1

'facebook'	12.5	300	2+3j	[1,2,3]
------------	------	-----	------	---------

0 1 2 3 4

→

Positive indexing

tuple

Collection of different data types.

Immutable, ordered

Items are separated with comma and enclosed in ().

Ex. t = (12,10.5,'hello')

dictionary

Unordered set of key-value paires.

Key has primitive datatype and value has any datatype.

Key-value paires are separated by comma.

Ex. d = {sachin:100 , kohli:90, dhoni:80}

set

Unordered collection of unique items.

Items are written inside { } and separated by commas.

Items are not in order.

Ex. a = {1,2,3,4,"hello"}

range

It gives immutable sequence of numbers betn start and stop.

Syntax:- range(start,stop,step)

Default starting is 0 and step is 1.

Ex. range(0,11) -> 0,1,2....10

None

None datatype means an object
that doesn't contain any value.

Ex. `a = None`

bool

Two values, True and False.

Used to determine given statements are true or false.

True is non-zero and False is 0.

Ex. a = True

Q.1 what is the output of following?:-

```
print(type("35"))
```

- 1) <class 'float'>
- 2) <class 'str'>
- 3) <class 'int'>

Q.1 what is the output of following?:-

```
a = {1,2,2,3,4,3}  
print(a)
```

- 1) {1,2,3,4,3}
- 2) {1,2,2,3,4,3}
- 3) {1,2,3,4}

Q.1 what is the output of following?:-

```
print(11,12,13,14)
```

- 1) 11,12,13,14**
- 2) 11 12 13 14**
- 3) (11,12,13,14)**

String in Python:-

- A string is a sequence of characters. Ex. “shantanu”
- Anything enclosed inside single quotes ,double quotes except escape sequences is [→]string.
- String is a sequence of unicode characters.

Creating string:-

‘hello’

“hello”

```
print('hello')  
print("hello")
```

Output:- hello
hello

- Triple quotes are also used but for multi-line strings and docstrings.

```
print("""hello,welcome to  
world of python""")
```

Accessing substring (characters) in string:-

Two ways:-

- 1) Indexing
- 2) Slicing

Indexing:- used to access individual(single) element of string.

slicing:- used to access group of elements of a string. We can access range of characters using slicing.

Accessing substring (characters) in string:-

Two ways:-

- 1) Indexing
- 2) Slicing

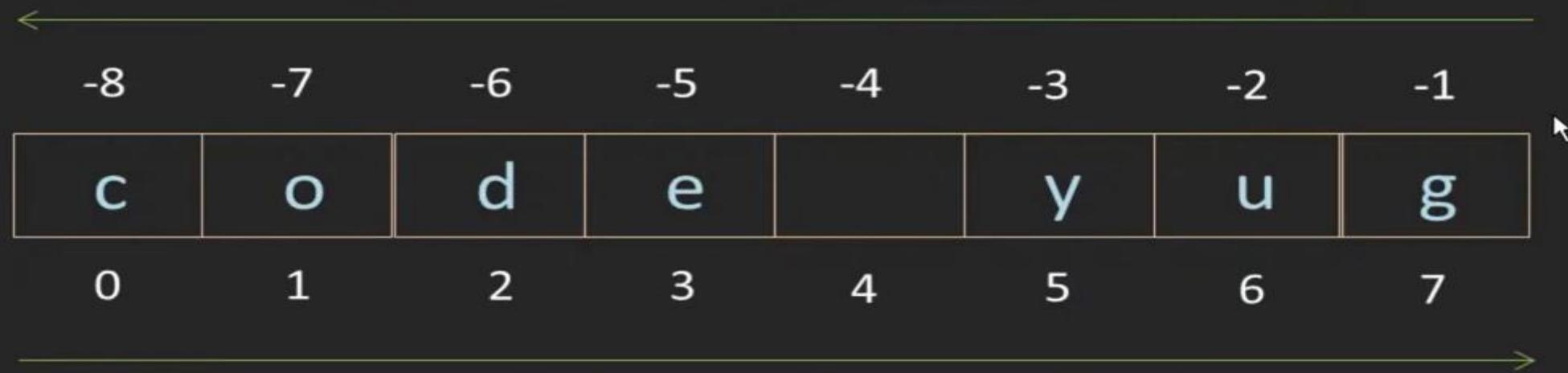
Indexing:- used to access individual(single) element of string.

slicing:- used to access group of elements of a string. We can access range of characters using slicing.

Index:-

- In python, Every element is represented by **index**.
Name = “code Yug”

Negative indexing / Reverse Indexing

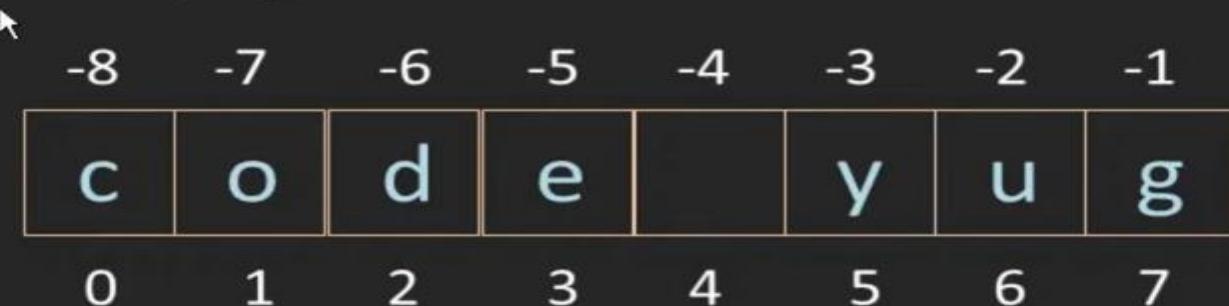


Positive indexing / Forward Indexing

Accessing Single Element:-

- To access single element, python uses indexing.

Name = “code Yug”



Syntax:- var_name[index]

Name[0] ---> c

Name[2] ---> d

Name[5] ---> y

Name[-1] ---> g

Name[-4] --->

Name[-8] ---> c

slicing:-

- To access a range of items (sub-string), we will use 'slicing'.
- You can return range of characters using slicing.
- [:] is a slice operator.

Syntax:-

Var_name[start:stop:step]

- start :- starting point of substring.
- stop :- It indicated end of string.
- step :- difference between indexing.

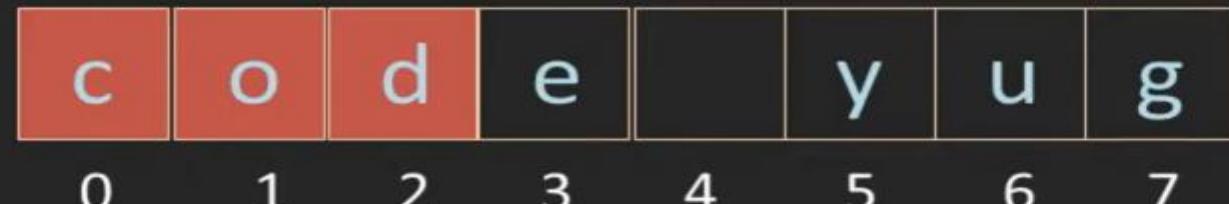
important:-

- start,stop,step are optional.
- Default start is index 0.
- Default step is 1 .
- end = stop -1
- Character at stop index is not printed.
- Positive index :- Go in forward direction
Negative index :- Go in Negative direction
- If stop not found,nothing is returned.

Python-Basics

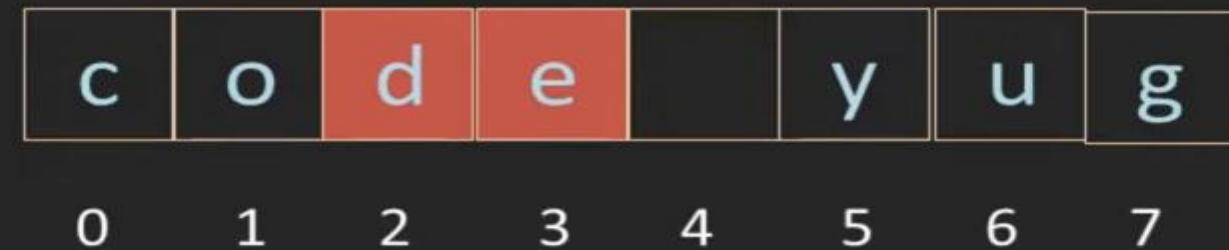
```
print(Name[0:3:1])      ----> cod
```

-8 -7 -6 -5 -4 -3 -2 -1



```
print(Name[2:4])      ----> de
```

-8 -7 -6 -5 -4 -3 -2 -1



Python-Basics

```
print(Name[0:5:2])      ----> cd_
```



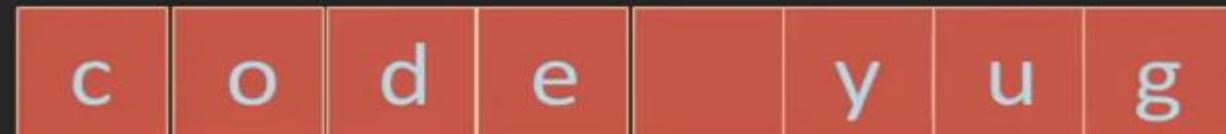
```
print(Name[:4])
```



Python-Basics

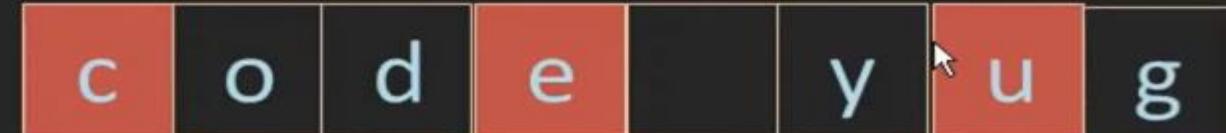
```
print(Name[:])      ----> code yug
```

-8 -7 -6 -5 -4 -3 -2 -1



```
print(Name[0::3])      ----> ceu
```

-8 -7 -6 -5 -4 -3 -2 -1



Python-Basics

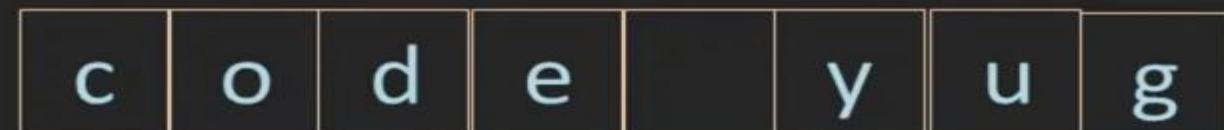
```
print(Name[-1:-4:-1]) ----> guy
```

-8 -7 -6 -5 -4 -3 -2 -1



```
print(Name[-1:-5:-2]) ----> gy
```

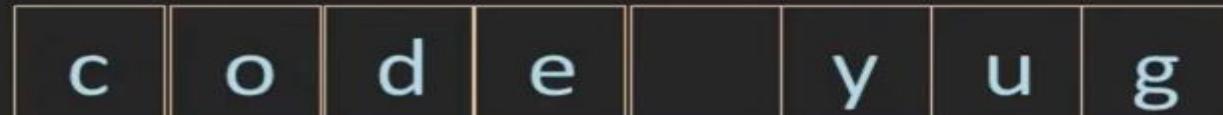
-8 -7 -6 -5 -4 -3 -2 -1



Python-Basics

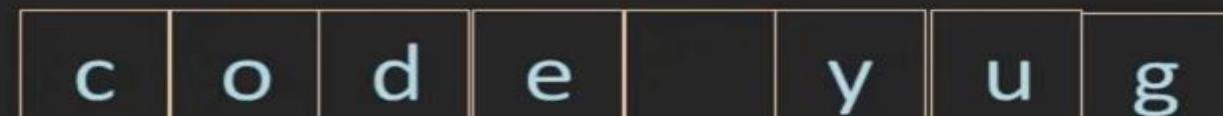
```
print(Name[-2:-6:2])
```

-8 -7 -6 -5 -4 -3 -2 -1



```
print(Name[-1: :-1])
```

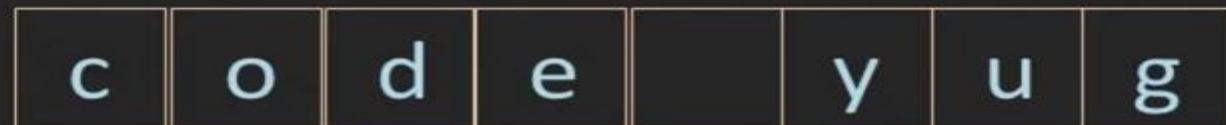
-8 -7 -6 -5 -4 -3 -2 -1



Python-Basics

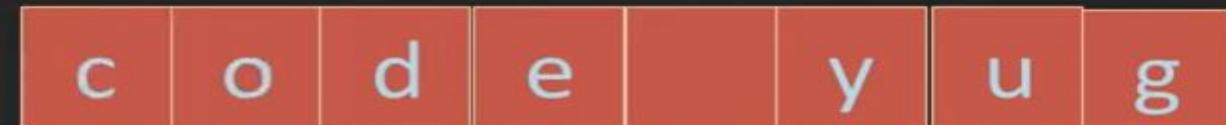
```
print(Name[-2:-6:2]) ----> "
```

-8 -7 -6 -5 -4 -3 -2 -1



```
print(Name[-1: :-1]) ----> guy edoc
```

-8 -7 -6 -5 -4 -3 -2 -1



Python-Basics

```
print(Name[-3:-11]) ----> "
```

-8 -7 -6 -5 -4 -3 -2 -1

c	o	d	e		y	u	g
0	1	2	3	4	5	6	7

```
print(Name[-1:-1:-1]) ----> "
```

-8 -7 -6 -5 -4 -3 -2 -1

c	o	d	e		y	u	g
0	1	2	3	4	5	6	7

```
string = "Hello, world!"  
print(string[7:12])  
a) world b) world! c) , worl d) , world
```

```
string = "Hello, world!"  
print(string[7:12])  
a) world b) world! c) , worl d) , world
```

```
string = "Hello, world!"  
print(string[:5])  
a) Hello b) Hello, c) , world! d) world!
```

```
string = "Hello, world!"  
print(string[:5])  
a) Hello b) Hello, c) , world! d) world!
```

```
string = "Hello, world!"  
print(string[-6:-1])
```

- a) world
- b) world!
- c) , worl
- d) , world

```
string = "Hello, world!"  
print(string[-6:-1])
```

- a) world
- b) world!
- c) , worl
- d) , world

Which of the following is the correct way to slice a string to get the last 3 characters?

- a) `string[-3:]`
- b) `string[3:]`
- c) `string[:-3]`
- d) `string[-3:-1]`

Which of the following is the correct way to slice a string to get the last 3 characters?

- a) `string[-3:]`
- b) `string[3:]`
- c) `string[:-3]`
- d) `string[-3:-1]`

Python-Basics- Q5

```
string = "Hello, world!"  
print(string[-5:2:-2])
```

- a) ol
- b) ow
- c) ,ol
- d) dlro

Python-Basics- Q6

```
string = "Python is fun!"  
print(string[::-2][::-1])
```

Which of the following slices would extract the substring "world" from the string "Hello, world!"?

- a) `string[7:12]`
- b) `string[-6:-1]`
- c) `string[7:-1]`
- d) All of the above

```
string = "Programming"
```

```
print(string[3:len(string) - 4])
```

S = [1,2,3,4,5,6,7,8,9]

print(s[:-4:-1])

- Concatenation of strings.
- Multiplication of string
- Deletion of string.
- Iteration through string
- Checking character/substring present in string.

concatenation:-

- Joining two or more strings into a single one.
- It adds strings together.
- The ‘+’ operator does this.
- Example:-

```
str1 = 'Python is'  
str2 = 'awesome'  
str3 = str1 + str2  
print(str3)
```

Output:-
Python isaweson

Multiplication:-

- Repeat same string for multiple times.
- Using ‘*’
- Example:-

```
str1 = 'codeyug'  
str2 = str1*2  
print(str2)
```

Deletion:-

- We cannot delete characters of string. But, deleting entire string is possible using ‘del’ keyword.
- Example:-

```
str1 = 'codeyug'  
del str1  
prints (str1)
```

Python-Basics- Q9

- Iteration on string
- Finding length of string
- len() function
- String membership
- Finding number of vowels in entered string.