Date.....
Page.....

Python for machine learning and data-science:-

↓

"Basic language"

- more easier than c++ and java
- faster

Big Companies using python:-

- Google → c++ and python for searching algorithm
- Netflix → Numpy and Sci-fy for numerical Analysis

fundamentals:-

case sensitive
have command terminator
semicolon for multistatement in one line

// use "#" for comment

Token:-

↳ Python break every logical-line into a set of lexical statements known as tokens

Sum of two numbers

```
a = 10
b = 20
print(a+b)
```

10 and 20 literals

a and b are variable (identifiers)

print → keyword

'+' → operator

Tokens

- Literals
  - ↓
  - 10, 22
  - 'abc', etc
  - ↓
  - ~~string = 'abc'~~
  - 22, 10 → numeric
  - Boolean = True, false
  - Special = none
  - Collection: sets, tuples, list, dictionaries
- operators
  - +, -, /, *, etc
- Identifiers
  - ↓
  - unique names
- Keywo
  - if, else, while, print

Operators in python:-

Arithmetic: +, -, /, *, %, //,
Comparision: <, >, !=
Bitwise operator: &, |, >>, <<, ~,
Membership: in, not in
Assignment: =, +=, -=, *=, /=
Logical: and, or
Identity: is, is not

Swapping two variable

```
a=10
b=20
print(a,b)
a=a+b      #30
b=a-b      #10
a=a-b      #20
print(a,b)
```

} Values changing

Option 2

```
a=10
b=20
print(a,b)
a=a^b
b=b^a
a=a^b
print(a,b)
```

Operators in python in details: (~~Bitwise~~)
Arithmetic

6 + 5 = 11
20 - 12.5 = 7.5
3.14 * 6 = 18.84

8 / 4 = 2.0
8 // 4 = 2
2 ** 5 = 2^5 = 32
10 % 3 = 1

~~Order of precedence: ** , - , *, /, //, %~~

Assignment operator:-

Operator	eg	Similar to
=	x = 10	x = 10
+=	x += 5	x = x + 5
-=	x -= 5	x = x - 5
*=	x *= 5	x = x * 5
/=		
%=		
**=		
&=		
^=		
<<=		

Comparision operator :-

outputs a boolean value

[True, false]

a = 3, b = 4

a == b	false
a != b	True
a > b	false
a < b	True
a >= b	False
a <= b	True

Logical operators :

and or not

x<5 and x<10 } Returns true if both statemen are true

or → Returns true if one Statement is true

not → Reverse the result

Identity operators:-

is / is not

↳ Returns true if both variable are same object

```
x is y
```

is not

↳ Returns true if both are not same

```
x is not y
```

Membership operators:-

in	True if	x in y
not in	true if	x not in y

Bitwise operators

↓

operations performed on binary digits.

1) Bitwise and

```
a = 0b1010          1 0 1 0
b = 0b1100          1 1 0 0
c = a&b             1 0 0 0
```

```
a = 10
b = 5
bin (a&b)
'0b01'
```

↳ Binary Equivalent.

Bitwise OR (|)

Returns 1 if any bit is 1

a = 10

b = 5

a = 1010

b = 0101

1111 → [15] → {decimal}

Bitwise Not (~)

↓

Return's one complement of a number

for eg: a = 10

↓

1010

~a = ~1010

⇒ 0101

⇒ 5 in decimal

Bitwise XOR (^)

Return 1 if one of the bit is one else it return 0.

a = 10 → 1010
b = 9 → 1001

0011 → 3

a^b ⇒ 3 }

2	10	0
2	5	1
2	2	0
	1	

a = 10 } → 1
b = 20 }

2	20	0
2	10	0
2	5	1
2	2	0
	1	

10100
10100
0000

10100
10100
11110

Data types in Python:-

Numbers, strings, list, tuples, dictionaries,

Sets

int
float
complex

" "

Basic string operations:-

my-string = "Good morning! Everyone.."
str-a = "I like to learn python"
str-b = "machine learning"
str-c = "Deep learning"

#indexing
print(str-a[0])

# slicing
print(str-c[0:4])
print(my-string[5:-2])

# Concatenation:-
print(my-string + str-a)

# repetation
print(str-b * 3)