Date ___/___/_____ Page No.: _____

Print function

print ('hello world')

or

print ("hello world")

output - hello world

string : collection of character inside "double quotes" or 'single quotes'

More things to know :

You can use 'single quotes' inside "double quotes".

or

"Double quotes" inside 'single quotes'.

ex:- print (" hello 'world' world")

output - hello 'world' world

print ('hello "world" world')

output - hello "world" world

Scanned by CamScanner

print ("hello 'world' world")

output : syntax error : Invalid syntax

print (' I'm Sangam')

output : syntax error : Invalid syntax

print (" I'm Sangam")

output : I'm Sangam

print (" Hello "world" world")

output : syntax error : Invalid syntax

Scanned by CamScanner

Escape Sequences

print ("hello \" world \" world")

output : hello "world" world

	Escape sequences	Meaning
1.	\'	single quote
2.	\"	double quote
3.	\\	backslash
4.	\n	new line
5.	\t	tab
6.	\b	backspace

ex:- print ('I\'m Sangam')

output : I'm Sangam

- \n -

print ("line A")
print ("line B")

output : line A
line B

[two different lines]

Scanned by CamScanner

Date / / Page No.:

print (" line A \n line B ")
output : line A
line B

print (" line A \n line B \n line C ")
output : line A
line B
line C
because of this

- \t -

print ("name\tSSangam")

output - name SSangam

- \\ -

print (" this is \ backslash")

output: this is \ backslash

print (" this is backslash\")

Output :
syntax error : End of liner [EOL]
while scanning string
literal

Scanned by CamScanner

Date __/__/____ Page No.: _____

print(" this is backslash \\")

Output : this is backslash \

print(" this is double backslash \\\\")

output : this is double backslash \\

- \b -

print("hell\bo")

output : helo

print("hell\blo")

output : hello

Scanned by CamScanner

Date ___/___/_____ Page No.: _____

Comments in Python

this is comment

print("lineA\n lineB") # \n is for new line

\t is for tab

print("name\t Sangam")

print("this is double backslash \\\\") # \\ is for backslash

symbol is used for comments.

(# _ _ _ _ _ _ _ _)

Scanned by CamScanner

Date __/__/__ Page No.:

Escape Sequences as Normal text

print ("Line A \\n Line B")
→ output - Line A \n Line B
print ("Line A \\t Line B")
→ output - Line A \t Line B
print ("this is 4 backslash \\\\\\\\")
→ output - this is 4 backslash \\\\

Q:- output - " '

Ans - print (" \" \' ")

Q:- output - \" \'

Ans:- print (" \\\" \\\' ")

\'	-	'
\"	-	"
\\	-	\

Scanned by CamScanner

Exercise

Q:- this is double backslash \\

Ans:- print ("this is double backslash \\\\")

Q:- this is /\/\/\ /\/\ mountain

Ans:- print ("this is /\\ /\\ /\\ /\\ /\\ mountain")

Q:- he is awesome

Ans:- print ("he is \t awesome")

Q:- \" \n \t \'

Ans:- print (" \\\" \\n \\t \\\'")

Scanned by CamScanner

Raw Strings

print (r" line A \n line B")

output : line A \n line B

print (r" line A \" line B")

output : line A \" line B

Python as a Calculator

print (2+3)
output - 5
print (2+3*4)
output - 14

Operators	Description	Example
+	addition	>>>print (2+3) 5
−	Substraction	>>> print (2-3) −1
*	Multiplication	>>> print (2*3) 6
/	Float division	>>> print (4/2) 2.0

Scanned by CamScanner

```
>>> print (2/4)
0.5
```

//	Integer division	`>>> print (4//2)`
		`2`
		`>>> print (2//4)`
		`0`

%	Modulo, it gives remainder	`>>> print (6%2)`
		`0`
		`>>> print (7%2)`
		`1`

**	Exponent	`>>> print (2**3)` `8`

In Python - Integer and float are data types.

Integer - Numbers without decimal point

Float - Numbers with decimal point

Scanned by CamScanner

Date ___/___/_____ Page No.: _____

```
print (2/4)          floating Point division
    0.5
print (4/2)
    2.0

print (4//2)         integer division
    2
print (2//4)
    0

print (2**3)
    8

print (2**0.5)   —   (√2 - 1.414)
    1.4142135623730951

round (2**0.5, 4)
print (round (2**0.5,4))
    1.4142

print (2**3/2*6-4*(3-4/2))

    (2**3/2*6-4*1)
    (8/2*6-4)
    (4.0*6-4)
    24.0-4
    20.0
```

Scanned by CamScanner

Date __/__/____ Page No.:

Precedence Rule

Operators	Precedence and Associativity rule
1. Parenthese	Highest
2. Exponent	Right to Left
3. *, /, //, %	Left to right
4. +, -	Left to right

What is Precedence rule?

To evaluate complex expression, python follows the rule of precedence, it governs the order in which the operation take place.

```
print((2+3)*5/2%6)
```

```
5*5/2%6
25/2%6
12.5%6
0.5
```

Scanned by CamScanner

exponents

```
print(2**3**2)

(2**9)
512
```

Scanned by CamScanner

Variable in Python

```
number1 = 2
print (number1)
    2
number1 = 4
print (number1)
    4
```

String and number can be stored in variable.

```
name = "Sangam"
print (name)
    Sangam

name = 123
print (name)
    123
```

So python is called as dynamic programming language.

Dynamic Programming language:

Dynamic Programming language, in computer science, is a class of high-level programming languages which, at runtime, execute many common programming behaviours that

"It is the prime responsibility of every citizen to feel that his country." —Sardar Vallabhai Patel

Scanned by CamScanner

Date __/__/____ Page No.:

static programming languages perform during compilation.

Rules

1. 1number = 4 X
   Syntax error
   variable never starts from number
   number1 = 4 ✓

2. First letter of variable
   (1) letter
   name = 'Sangam' ✓
   (2) Underscore- _name = 'khare' ✓

3. name$ = 'SK' X
   $usd = 34 X [Syntax error]
   u$sd = 34 X
   special symbol never be used in variable.

4. u1sd = 34 ✓
   us3d = 36 ✓
   us9f = 37 ✓

Scanned by CamScanner

Date / / Page No.:

Conventions

useronename = 'Sangam' ✓

user_one_name = 'Sangam' ✓

(snake case writing)

[Usable in Python]

userOneName = 'Sangam' ✓

(camel case writing)

[Usable in Programming language like as Java]

We can use any writing style which are shown above.

But according to Google's style guide for python we should use snake case writing in Python

Scanned by CamScanner

String Concatenation

```
first_name = "Sangam"
last_name = "khare"

full_name = first_name + last_name
print (full_name)
```

output - Sangamkhare

```
full_name = first_name + " " + last_name

print (full_name)
```

output - Sangam khare

or

```
first_name = "Sangam"
last_name = "khare"

full_name = first_name + last_name
print (full_name)
```

output - Sangam khare

We can not add numbers with string.

ex -
```
first_name = "Sangam"
print (first_name + 3)
```

output : error - type error

Scanned by CamScanner

Date __/__/__ Page No.:

But we can add string with string.

print (first_name + "3")

output - Sangam3

<u>or</u>

print (first_name + str(3))

output - Sangam3

* -

print (first_name * 3)

output - SangamSangamSangam

first_name = "Sangam"

print (first_name * 3)

output - Sangam Sangam Sangam

Scanned by CamScanner

Date ___/___/___ Page No.: ___

User Input

Input function is used for User Input

```
name = input("type your name")
```

output - type your name Sangam

```
print("hello" + name)
```

output - type your name Sangam
Hello Sangam

Note - Input function takes input from user always in string.

ex-
```
age = input("What is your age")
print("your age is " + age)
```

output -
What is your age 24
your age is 24

[input function taken 24 as "24"]

Scanned by CamScanner

Date __/__/____ Page No.: ____

int() function

```
number_one = input ("enter first number")
number_two = input ("enter Second number")

total = number_one + number_two

print ("total is " + total)
```

output -

```
enter first number5
enter Second number6
total is 56
```

[because of "5" + "6" = 56]

int() →

```
num1 = int(input ("enter no. "))
num2 = int(input ("enter no. "))

total = num1 + num2

print ("total is " + total)
```

output -

```
enter no. 2
enter no. 3
Type error : must be str, not int
```

[because of "total is" + 5]

Scanned by CamScanner

```
num1 = int(input("enter no. "))
num2 = int(input("enter no. "))

total = num1 + num2

print("total is " + str(total))
```

output - enter no. 8
enter no. 5
total is 13

[because of str(13) = "13"]

String function

ex:- str(4) → "4"
str(6) → "6"
str(3.0) → "3.0"

int function

ex:- int("4") → 4
int("6") → 6
int(3.0) → 3

float function

ex:- float("4") → 4.0
float("6") → 6.0
float(3) → 3.0

Scanned by CamScanner

Date ___/___/_____ Page No.:

Exercise

```
number1 = str(4)
number2 = float("44")
number3 = int("33")

print(number2 + number3)
```

44.0 + 33

output - 77.0

[we can add int with float but resultant will be float]

[float + int = float]

Scanned by CamScanner

Date __/__/____ Page No.: ____

More about Variables

name, age = "Harshit", "24"

[name = "Harshit"
age = "24"]

print ("hello " + name + " your age is " + age)

output :
hello Harshit your age is 24

ex-

x = y = z = 1
print (x+y+z)

[Multiple variable declare in one line]

output : 3

Scanned by CamScanner

Date __/__/____ Page No.: ____

Two or More Input in one line

```
name = input ("enter your name : ")
age  = input ("enter your age : ")
```

output -

```
enter your name : Sangam
enter your age : 18
```

```
name, age = input ("enter your name and age"). split ()
```

[means, space between inputs]

```
print (name)
print (age)
```

output :

```
enter your name and age Harshit 24
Harshit
24
```

```
name, age = input ("enter your name and age "). split (",")
```

[means, comma between inputs]

```
print (name)
print (age)
```

output : enter your name and age Sk, 24

```
Sk
24
```

"It is the prime responsibility of every citizen to feel that he contributes ...

Scanned by CamScanner

String formatting

```
name = "Sangam"
age  = 24

print("hello " + name + " your age is "
        + str(age))
                        # ugly syntax
```

String formatting

1. Python 2 — X [not usable in modern time]
2. Python 3
3. Python 3.6 [Best syntax]

{} → Place holder

Python 3 syntax :

```
print("hello {} your age is {}".format
        (name, age))
```

Python 3.6 syntax :

```
print(f"hello {name} your age is {age}")
```

Output :
hello Sangam your age is 24

We can also use like this :

```
print("your age is {}".format(age+2))
print(f"your age is {age+2}")
```

[+, -, *, /]
any calculation

Scanned by CamScanner

Output: your age is 26

Scanned by CamScanner

Page No.

Date ___/___/___

Exercise

Ask user to input 3 numbers and you have to print average of three numbers using string formatting.

Bonus: try to take all three comma separated inputs in one line.

```
n1, n2, n3 = input("enter 3 numbers separated with commas : ").split(",")

print(f"average = {(int(n1)+int(n2)+int(n3))/3}")
```

output:
enter 3 numbers separated with commas : 3,3,3
average = 3.0

```
n1 = input("enter no.: ")
n2 = input("enter no.: ")
n3 = input("enter no.: ")

print(f" average = {(int(n1)+int(n2)+int(n3))/3}")
```

output:
enter no.: 5
enter no.: 6
enter no.: 4
average = 5.0

Scanned by CamScanner

Date __/__/____

String Indexing

```
language = "python"
# positions (index number)
          index     index
# p = 0   ,  -6
# y = 1   ,  -5
# t = 2   ,  -4
# h = 3   ,  -3
# o = 4   ,  -2
# n = 5   ,  -1

print(language[0])
print(language[1])
print(language[2])
print(language[3])
print(language[4])
print(language[5])
```

Output - p
y
t
h
o
n

Scanned by CamScanner

index number in square brackets
- [index number]

ex- i[5]

language = "Java"

print(language[4])

output → Index error : string index out of range

language = "Python"

print(language[-1])

output - n

if we dont know, which is the last character (and dont know how many character in string) then we use [-1] index of string.

print(language[-4])

output : t

print(language[-8])

Index error : string index out of range

Scanned by CamScanner

Date __/__/____ Page No.: ____

String Slicing

Syntax:
Variable [start argument : stop argumen

```
lang = "python"
print (lang [0:2])
```

output : py

```
print (lang [0:6])
```

output : python

```
print (lang [2:4])
```

output : th

```
print (lang [3:6])
```

output : hon

```
print (lang [-3:6])
```

output : hon

```
print (lang [ : ])
```

output : python

Scanned by CamScanner

Date __/__/____ Page No.: _____

print(lang[1:])

output : ython

print(lang[2:])

output : thon

print(lang[:3])

output : pyt

Step Argument

Variable[Start argument : stop argument : step argument]

print("Harshit"[2:5])

output : rsh

print("Harshit"[0:5])

output : Harsh

print(lang[0:4:1])

output : pyth

print("Harshit"[0:5:1])

output : Harsh

Scanned by CamScanner

Page No.:

Date / /

```
print("Harshit"[0:5:2])
output: Hrh
print("Harshit"[0::2])
output: Hrht
print("Harshit"[0::3])
output: Hst
print("Harshit"[0::2])
output: Hrht
print("Harshit"[::3])
output: Hst
print("Harshit"[5::-1])
output: ihsraH
print("Harshit"[-1::-1])
output: tihsraH
```

or

```
print("Harshit"[::-1])   # trick
output: tihsraH
```

Scanned by CamScanner

Date __/__/____ Page No.: _____

Exercise

Q.- Ask user name and print back user name in reverse order.

Note: try to make your program in 2 lines using string formatting.

```
name = input ("enter your name: ")
r = name[-1::-1]
print(f" reverse = {r}")
```

(in 3 lines)

output : enter your name: Sangam
reverse = magnaS

or

```
print(f"reverse = {name[-1::-1]}")
```

output : enter your name : Sangam
reverse = magnaS

Scanned by CamScanner

String Methods

name = "Sangam khare"

1. len() function

print(len ("Sangam"))

output : 6

print(len (name))

output : 12

length = len (name)

print(length)

output : 12

2. lower () method

name = "HarShiT vAshisHThA"

print(name.lower ())

output : harshit vashishtha

Scanned by CamScanner

Date __/__/____ Page No.: ____

```
L = name.lower()
print(L)
```

output : harshit vashishtha

3. upper() method

```
name = "sAngaM"
print(name.upper())
```

output : SANGAM

```
U = name.upper()
print(U)
```

output : SANGAM

4. title() method

```
name = "sAngam"
print(name.title())
```

output : Sangam

```
name = "saNgaM kHAre"
print(name.title())
```

output : Sangam Khare

Scanned by CamScanner

Date ___/___/_____ Page No.:

```
name = "sk rk uk"
print(name.title())
```

output : Sk Rk Uk

5. count() method

```
name = "Sangam"
print(name.count("m"))
```

output : 1

```
print(name.count("a"))
```

output : 2

```
name = "Harshit"
print(name.count("H"))
```

output : 2

```
print(name.count("t"))
```

output : 1

```
name = "Sangam khare"
print(name.count("a"))
```

output : 3

"It is the prime responsibility of every citizen to feel that ...

Scanned by CamScanner

Date __/__/____ Page No. _____

Exercise

Take two comma separated inputs from user

1. user's name
2. single character

output : 2 print lines
1. user's name length
2. count the character that user inputed
(bonus : case insensitive count)

```
name, char = input ("enter comma separated
                name and character: ").split(",")

print (f"length of your name is {len(name)
                }")

print (f" character count : {name.count
                (char)}")
```

output :
enter comma separated name and
character : Harshit, h

length of your name is 7
character count : 0 [because it reads 'space' not h so output is 0]

"Education is the manifestation of perfection already in a man." – Swami Vivekananda

Scanned by CamScanner

Output :
enter comma separated name and character : Sangam, a

length of your name is 6
character count : 2

output :
enter comma separated name and character : Harshit, h

length of your name is 7
character count : 1
(only count h)
[not H]

Case insensitive →

```
name, char = input("enter comma separated name and character : ").split(",")

print(f"length of your name is {len(name)}")

print(f"character count : {name.lower().count(char.lower())}")
```

output :
enter comma separated name and character : Harshit, h

length of your name is 7
character count : 2

Scanned by CamScanner

Date ___/___/_____ Page No.: _____

Solve Problem with spaces using strip method

```
name = "    Harshit    "
dots = "...................."
```

for removing left side space - lstrip() method

for removing right side space - rstrip() method

```
print (name)
```

output : Harshit

```
print (name + dots)
```

output : Harshit .....

```
print (name.lstrip() + dots)
```

output : Harshit .....

```
print ( name.rstrip() + dots)
```

Output : Harshit.....

```
print ( name.strip() + dots)
```

Output : Harshit.....

Scanned by CamScanner

Date __/__/____ Page No.: ____

```
name = "    Har    shit    "
dots = "..............."

print (name.strip() + dots)
```

output : Har shit.....

If you want to remove this space then use following method.

```
name.replace(" ", "")
```

space

no space

replace with

```
print (name.replace(" ", ""))
```

output : Harshit

```
print (name.replace(" ", "") + dots)
```

output : Harshit.....

Scanned by CamScanner

Date __/__/____ Page No: _____

```
name, char = input("enter comma separated
            name and character:").split(",")

print(f"length of your name is
                {len(name)}")

print(f"character count : {name.strip().
    lower().count(char.strip().lower())}
                ")
```

output:

```
enter comma separated name and
            character :  Harshit , h

length of your name is 18
character count : 2
```

Scanned by CamScanner

Date __/__/____ Page No.:

Find and Replace method

replace() method
find () method

Replace() method

string = "she is beautiful and she is good dancer"

print(string.replace(" ", "_"))

output : she_is_beautiful_and_she_is_good_dancer

string = " He is handsome and he is good dancer."

print(string.replace("is", "was"))

Output: He was handsome and he was good dancer.

print(string.replace("is", "was", 1))

output: He was handsome and he is good dancer.

(only one "is" replace with "was")

Scanned by CamScanner

Date __/__/____ Page No.: _____

print(string.replace("is", "was", 2))

output : He was handsome and he was good dancer.

(two "is" replace with "was")

print(string.replace("is", "was", 3))

output : He was handsome and he was good dancer.

(no difference in output)

find() method

string = "she is beautiful and she is good dancer"

print(string.find("is"))

output : 4

string = "is she is beautiful"

print(string.find("is"))

output : 0

print(string.find("is", 1))

(searching start from this position)

output : 7

Scanned by CamScanner

string = " she is beautiful and
she is good dancer"

(If we dont know first "is" position and want to find second "is" position) then :

is_position1 = string.find ("is")

is_position2 = string.find ("is", is_position1 +1)

print (is_position2)

output : 25

[is_position1 : 0]

[is_position1 +1 = 4 + 1 = 5
because if we want to find second "is" position then we start finding position from first "is" position +1]
[neither first "is" position found at 0th position and not get second "is" position]

Scanned by CamScanner

Date __/__/__ Page No.:

Center method of String

```
name = "harshit"
print(name.center(9, "*"))
```

[length of harshit - 7
** - 2] 9

output : *harshit*

```
print(name.center(8, "*"))
```

output : harshit*

```
print(name.center(11, "*"))
```

output - : **harshit**

ex-
```
name = input("enter your name : ")
print(name.center(len(name)+8, "*"))
```

output : enter your name : Sangam
****Sangam****

```
name = 'harshit'
print(name.center(10, "*"))
```

output : *harshit**

Scanned by CamScanner

Date __/__/____

Strings are immutable

```
string = "string"
print(string[1])
```

output : t

if we want to change the character :

```
string[1] = 'T'
```

output : string[1] = 'T'
TypeError : 'str' object does not support item assignment

Thats why Strings are immutable

[If we once made string, we can not change it]

```
String = 'string'
print(string.replace('t','T'))
```

output : sTring

Scanned by CamScanner

Date __/__/____

- string = 'string'

  string.replace ('t', 'T')

  print (string)

  output : string

- string = 'string'

  new = string.replace ('t', 'T')

  print (new)
  print (string)

  output : sTring
  string

  (If we once defined string we can not change it)

  [If we want to change we have to create another variable to assign another string]

  But original string will remain unchange.

Scanned by CamScanner

Assignment Operators

```
name = 'harsh'
name = name + 'it'
                [name += 'it']
print (name)
```

output : harshit

Note : We might think how can we change 'name' variable if strings are immutable?

Remember one thing, in python we can change value of variable anytime

but other methods cannot change our string or we cannot change particular character from string.
(immutability)

```
age = 23
age = age + 1
                [age += 1]      age*=1
print (age)
```

Output : 24

Output : 23

Scanned by CamScanner

Page No.:

[age -= 2]

output : 21

[age * = 2]

output : 46

Scanned by CamScanner

If statement

syntax :

```
if condition :
    code # if condition is tru
         then this code will
         execute
code
```

```
age = input ("enter your age : ")
age = int (age)
if age >=14:
    print (" you are above 14")
```

output : enter your age : 15
you are above 14

output : enter your age : 13

-
```
age = input ("enter your age : ")
age = int (age)
if age >= 14:
{ print (" line a")
{ print (" you are above 14")
```

indentation

Scanned by CamScanner

Date __/__/____

output : enter your age : 16
line a
you are above 14

```
age = input("enter your age : ")
age = int(age)
if age >= 14:
print("line a")
print("you are above 14")
```

(No indentation)

output : [enter your age : 16]

Indentation error: expected an indented block

```
if age >=14:
    print("Yes")
    print("eligible")
```

space

Indentation

Yes
eligible

In other languages

```
if age >=14
{ print("Yes")
  printf("eligible")
}
```

Yes
eligible

Scanned by CamScanner

Date / /

Page No.:

Pass Statement

```
x = 18
if x>18:
```

Output: SyntaxError: unexpected EOF while parsing

```
x = 18
if x>18:
    pass
```

[If you dont want to (write anything) write statement then you simply write pass]

Scanned by CamScanner

Date / /

If - else statement

```
age = input ("enter your age : ")
age = int (age)
if age >=14:
    print ("line a")
    print ("you are above 14")

else:
    print ("sorry, you can't play ")
```

output : enter your age : 13
sorry, you can't play

output : enter your age : 14
line a
you are above 14

Scanned by CamScanner

Exercise

Number Guessing Game

Make a variable, like winning_number and assign any number to it.

Ask user to guess a number.

if user guessed correctly then print "You win!!!!"

if user did'nt guessed correctly then:

- if user guessed lower than actual number then print "too low"
- if user guessed higher than actual number then print "too high"

google "how to generate random number in python" to generate random winning number

Scanned by CamScanner

Date __/__/____

(using nested if-else)

```
winning_number = 27
user_input = input("guess a number b/w
                    1 and 100")

user_input = int(user_input)

if user_input == winning_number:
    print("You Win!!!!")

else:
    if user_input < winning_number:
        print("too low")

    else:
        print("too high")
```

Output :

```
guess a number b/w 1 and 100 37
too high

guess a number b/w 1 and 100 21
too low

guess a number b/w 1 and 100 27
You Win!!!!
```

Scanned by CamScanner

Date ___/___/___ Page No.: ___

And, Or Operator

```
# check two conditions at same time

# and, or
# and operator
name = 'abc'
age = 19

if name == 'abc' and age == 19:
    print("Condition true")

else:
    print("Condition false")
```

output : Condition true

```
if name == 'abc' and age == 23:
    print("True")
else:
    print("False")
```

output : False

```
if name == 'abcd' and age == 19:
    print('True')
else:
    print('False')
```

Scanned by CamScanner

Date __/__/____

Output : False

```
if name == 'SK' and age == '18:
        print('True')
else:
        print ('False')
```

output : False

Or operator

```
name = 'abc'
age = 19

if name == 'abc' or age == 19:
        print (" True")

else:
        print (" False")
```

Output : True

```
if name == 'abcd' or age == 19:
        print ('True')
else:
        print ('False')
```

output : True

Scanned by CamScanner

```
if name == 'abc' or age == 16:
        print ('True')
else:
        print ('False')
```

output : True

```
if name == 'abcd' or age == 10:
        print ('True')
else:
        print ('False')
```

output : False

Scanned by CamScanner

Date ___/___/_____ Page No.:

Exercise

Watch coco

Ask user's name and age

If user's name start with ('a' or 'A')
and age is above=10 then

print 'you can watch coco movie'

else
print 'sorry, you can't watch coco'

```
name = input ('Enter your name: ')
age  = input ('Enter your age:')

age = int (age)

if age >=10 and (name[0] == 'a' or
                 name[0] == 'A'):
    print ('you can watch coco movie')

else:
    print ('sorry, you can't watch coco')
```

Scanned by CamScanner

Page No.

Date

if.... elif.... else Statement

```
# show ticket pricing
  1 to 3    (free)
  4 to 10   (150)
  11 to 60  (250)
  60 above  (200)

age = input("please input your age : ")

age = int(age)

if 0<age<=3:
        print("Ticket price : Free")

elif 3<age<=10:
        print(" Ticket price : 150")

elif 10<age<=60:
        print(" Ticket price : 250")

else:
        print(" Ticket price : 200")
```

```
elif age==0 or age<0:
        print("you can't
                watch")
```

Scanned by CamScanner

Date __/__/____ Page No.: ____

syntax :

```
if   test - condition :
        ____
        ____
        ____

elif  test - condition :
        ____
        ____
        ____

else:
        ____
        ____
        ____
```

Scanned by CamScanner

Date ___/___/_____ Page No.: _____

In keyword

```
name = 'Harshit'

# in Keyword
# if with in
                    [if 'a' in 'Harshit']
if 'a' in name:
    print ("a is present in name")
else:
    print (" not present ")
output : a is present in name

if 'z' in 'Harshit':
    print ("z is present in name")
else:
    print (" not present ")

output : not present
```

Scanned by CamScanner

Check Empty or Not

```
name = "abc"

if name:  # true if string is not empty
        print("string is not empty")
else:
        print("string is empty")
```

output : string is not empty

```
name = ""

if name: # true if string is not empty
        print("string is not empty")
else:
        print("string is empty")
```

output : string is empty

```
name = input('enter your name :')

if name:
        print(f"your name is {name}")
else:
        print("you did'nt type anything")
```

Scanned by CamScanner

While loop

```
# loops
# while loop ,

print ("hello world") # 10 times

i = 1
while i <= 10:
    print ("hello world")
    i = i + 1
```

output : hello world
hello world
hello world
hello world
hello world
hello world
hello world
hello world
hello world
hello world

```
i = 1
while i <= 10:
    print (f"hello world {i}")
    i = i + 1
```

output : hello world 1
hello world 2
.....
hello world 9
hello world 10

Scanned by CamScanner

Sum of Numbers program using while loop :

```
total = 0
i = 1
while i <= 10:
    total = total + i
    i = i + 1
print(total)
```

output : 55

Scanned by CamScanner

Date ___/___/___ Page No.:

Exercise

Sum of n natural numbers
ask a user for natural number (n)
print total from 1 to n

```
n = input ("Enter a number: ")

n = int (n)

total = 0
  i = 1
while i<=n:
      total = total + i
      i = i+1
print (total)
```

output : Enter a number : 10
55

Enter a number : 20
210

Scanned by CamScanner

Date / /

Page No.

Exercise

Ask user to input a number containing more than one digit

example - 1256

calculate 1+2+5+6 and print

algorithm (method to solve problem in human language)

ask input in string, i.e. dont change string to int

example = "1256"

pick string character one by one and change to int

int (example[0]) + int (example[1]) . . . . .
. . . go upto
len(example)

Scanned by CamScanner

```
number = input("enter a number")

total = 0
  i = 0
while i < len(number):
    total = total + int(number[i])
    i += 1
print(total)
```

Exercise

Ask a user for name
Example - Harshit Vashisth
print count of each words

output :

```
H : 1
a : 2
r : 1
s : 3
h : 3
i : 2
t : 2
  : 1
V : 1
```

Scanned by CamScanner

Date __/__/__ Page No.: ____

Infinite Loop

```
i = 0
while i <= 10:
    print("hello world")
```

[if not increase the value of i, i = i + 1]

Output : hello world
hello world
hello world
hello world
. . . . . .
infinite loop

[for stop the infinite loop we have to press ctrl + c on the keyboard]

if we want to loop should be infinite then :

```
while True:
    print("hello world")
```

Output : hello world
hello world
. . . . . .
infinite loop

Scanned by CamScanner

In python True and False are Boolean.

Scanned by CamScanner

Date ___/___/_____ Page No.: _____

For loop

```
for i in range(10):        # 0 to 9(10 range
    print("hello world")
```

Output : hello world
hello world
hello world
hello world
hello world
hello world
hello world
hello world
hello world
hello world

```
for j in range(10):
    print(f"hello : {j}")
```

output : hello : 0
hello : 1
hello : 2
hello : 3
hello : 4
hello : 5
hello : 6
hello : 7
hello : 8
hello : 9

Scanned by CamScanner

Date / /

Page No.:

```
for num in range(11):        # range 0 to 10 (11-1)
    print(f"{num}")
```

output:
0
1
2
3
4
5
6
7
8
9
10

```
for num in range(1,11):      # range 1 to 10
    print(f"{num}")
```

output:
1
2
3
4
5
6
7
8
9
10

```
for i in range(1,4):         # range 1 to 3 (4-1)
    print("hey")
    print("Sangam")
```

Scanned by CamScanner

output : hey
Sangam
hey
Sangam
hey
Sangam

```
for i in range (1,4):      # 1 to 3
    print (" Hey")
    print ("\n good bye")
```

output : Hey

good bye
Hey

good bye
Hey

good bye

```
for i in range (1,3):      # 1 to 2
    print ("hey")
    print (" bro\n")
```

output : hey
bro

hey
bro

Scanned by CamScanner

Page No.:

Date / /

for loop example

Sum from 1 to 10

ex.-
```
total = 0
for i in range (1, 11):
    total = total + i

print (total)
```

output - 55

ex-
```
n = input (" enter the number : ")

n = int (n)

total = 0

for i in range (1, n+1):
    total = total + i

print (total)
```

output : enter the number : 10
55

output :
enter the number : 20
210

Scanned by CamScanner

for loop example

Ask user a number like 1256
calculate sum of digits → 1+2+5+6

```
total =0
num = input("Enter a number: ")
for i in range(0, len(num)):
    total = total + int(num[i])
print(total)
```

output : Enter a number : 1349
17

Scanned by CamScanner

for loop example

Ask user name and count each character

```
name = input("enter your name: ")
temp = ""
for i in range(0, len(name)):
```

or

```
for i in range(len(name)):
    if name[i] not in temp:
        print(f"{name[i]}:{name.count(name[i])}")
        temp = temp + name[i]
```

Scanned by CamScanner

Date ___/___/_____ Page No.: _____

Break and Continue keyword

```
for i in range (1,11):
        if i==5:
            break ———— execution stop
        print(i)
```

output : 1
2
3
4

```
for i in range (1,11):
        if i==5:
            continue
        print (i)
```

output : 1
2
3
4
6
7
8
9
10

Scanned by CamScanner

Exercise

Modify number guessing Game

```
winning_number = 43
    guess = 1
number = input("guess a number b/w
                1 and 100 : ")
number = int(number)

game_over = False

while not game_over:
    if number == winning_number:
        print(f"you win, and you guessed
            this number in {guess} times")
        game_over = True

    else:
        if number < winning number:
            print("too low")

            guess += 1
            number = int(input("guess
                again:"))
        else:
            print("too high")
            guess += 1

            number = int(input("guess
                again:"))
```

Scanned by CamScanner

DRY principle of coding

Date / / Page No.:

DRY : Don't Repeat Yourself

```
winning_number = 43
guess = 1
number = int(input("guess a number
                between 1 and 100:"))

game_over = False

while not game_over:
    if number == winning_number:
        print(f"you win, and you
            guessed this number in {guess}
            times")
        game_over = True

    else:
        if number < winning_number:
            print("too low")
        else:
            print("too high")

        guess += 1

        number = int(input("guess again:
```

[This is the best way to design game]

Scanned by CamScanner

Date __/__/____

```
winning_number = 43
guess = 1
number = int(input("guess a number
            between 1 and 100 : "))

game_over = False
                    [infinite loop]
while True:
    if number == winning number:
        print(f"you win and you
            guessed this number
            in {guess} times")
        break

    else:
        if number < winning_number:
            print("too low")
        else:
            print("too high")

        guess += 1
        number = int(input("guess again: "))
```

Scanned by CamScanner

if you want to random winning no.

```
import random
winning_number = random.randint(1, 100)
```

```
winning_number = 43
guess = 1
game_over = False

while True:
    number = int(input("guess a number between 1 and 100: "))
    if number == winning_number:
        print(f"you win, and you guessed this no. in {guess} times")
        break
    else:
        if number < winning_number:
            print("too low")
        else:
            print("too high")

    guess += 1
    continue
```

Scanned by CamScanner

Date ___/___/_____

Step Argument in range function

Stop argu.

Start argu. ← ↑ → Step argument

```
for i in range (1, 11, 1):
    print(i)
```

output : 1
2
3
4
5
6
7
8
9
10

```
for i in range (1, 11, 2):
    print(i)
```

output : 1
3
5
7
9

```
for i in range (0, 11, 2):
    print(i)
```

output : 0
2
4
6
8
10

"Education is the manifestation of perfection already in a man." –Swami Vivekananda

Scanned by CamScanner

Stop argu.

Start argu. ← ↑ → Step argu.

```
for i in range(10,0,-1):
    print(i)
```

Output:
```
10
9
8
7
6
5
4
3
2
1
```

```
for i in range(10,-1,-1):
    print(i)
```

output:
```
10
9
8
7
6
5
4
3
2
1
0
```

```
for i in range(+1,-11,-1)
    print(i)
```

output:
```
1
0
-1
-2
-3
-4
-5
-6
-7
-8
-9
-10
```

"It is the prime responsibility of every citizen to feel that his country."

Scanned by CamScanner

Date / /

for loop and string

```
name = 'harshit'
for i in range (len(name)):
    print (name[i])
```

output : h
a
r
s
h
i
t

This is for all progra. languages
ex:- javascript, python etc.

```
name = 'harshit'
for i in name:
    print (i)
```

output : h
a
r
s
h
i
t

This is for python

Because in Python strings are iterable

Scanned by CamScanner

```
num = input ("enter a number:")
total = 0
for i in num:
    total += int(i)
print (total)
```

Scanned by CamScanner

Date __/__/____ Page

Function Intro

```
def function_name (parameters):
    ========
    ========
    ========
    ========
    ========

print (function_name (arguments))
```

These are Parameters

```
def add_two (a,b):
    return a+b
```

These are arguments

```
L=add_two (5,4)

print (L)
```

output : 9

```
print (add_two (6,7))
```

output : 13

ex.-

```
a = int (input ("enter first number : "))

b = int (input ("enter second number : "))

total = add_two (a,b)

print (total)
```

output : enter first number : 30
enter Second number : 40
70

Scanned by CamScanner

ex:- first_name = input ("first name : ")

last_name = input ("last name : ")

print (add_two (first_name, last_name))

output : first name : Sangam
last_name : khare
Sangam khare

Python is the dynamic programming language.

because in this function we pass anything (two strings, two integers, two floating point numbers) two inputs, and it returns their sum.

Scanned by CamScanner

Date __/__/____

Return Vs Print in Functions

→ These are Parameters

```
def add_three (a,b,c):
    return a+b+c
```

→ These are arguments

```
L=add_three (5,5,5)
print (L)
```

or print(add_three(5,5,5))

output : 15

```
def add_three (a,b,c):
    print (a+b+c)

add_three (10,5,4)
```

output : 19

Scanned by CamScanner

Function Practice

ex.-
```
def last_char (name):
    return name[-1]
```
→ Parameter

```
print(last_char ('sangam'))
```
→ argument

output : m

```
last_char (9)        # error
```

output : TypeError : 'int' object is not subscriptable

ex.-
```
def odd_even (num):
    if num % 2 == 0 :
        return "even"
    else:
        return "odd"
```

```
print (odd_even (9))
```

output : odd

```
print (odd_even (10))
```

output : even

Scanned by CamScanner

Date __/__/____

ex:-
```
def odd_even (num):
    if num%2 ==0:
        return "even"
    return "odd"

print (odd_even (10))
```

output : even

```
print (odd_even (9))
```

output : odd

not

ex:-
```
def odd_even (num):
    if num%2 ==0:
        return "even"
    return "odd"
L = odd_even (7)
print (L)
```

output : odd

ex:-
```
def is_even (num):
    if num%2 ==0:
        return True
    else:
        return False

print(is_even (9))
```

Output : False

Scanned by CamScanner

parameter

ex.- def is_even (num):
    if num%2 == 0:
        return True

    return False

argument

print (is_even (8))

output : True

ex- def is_even (num):
    return num%2 == 0 # True or False

print (is_even (9))

output : False

print (is_even (10))

output : True

ex.- def song():

    return "happy birthday song"

print (song())

output : happy birthday song

Scanned by CamScanner

Exercise

Define a function which takes two numbers as a input and return greater of two numbers.

```
def greater (a,b):
    if a>b:
        return a
    else:
        return b

num1 = input ("enter first number : ")
num1 = int (num1)

num2 = input ("enter second number : ")
num2 = int (num2)

bigger = greater (num1, num2)

print (bigger)
```

or

```
print (f"{bigger} is greater")
```

output: enter first number : 12
enter second number : 14
14 is greater

Scanned by CamScanner

Define Greatest

```
def greatest (a,b,c):
    if a>b and a>c:
        return a
    elif b>a and b>c:
        return b
    else:
        return c

print (greatest (10,20,30))
```

output : 30

```
print (greatest (10,40,30))
```

output : 40

```
print (greatest (100,40,10))
```

output : 100

Scanned by CamScanner

Date __/__/__

Function inside Function

```
def greater (a,b):
    if a > b:
        return a
    return b

def greatest (a,b,c):
    if a>b and a>c:
        return a
    elif b>a and b>c:
        return b
    else:
        return c

def new_greatest (a,b,c):
    bigger = greater (a,b)
    return greater (bigger, c)
    [return greater
        (greater(a,b), c)]

print (new_greatest (10,20,30))
```

output : 30

```
print (new_greatest (12,15,9))
```

output : 15

Scanned by CamScanner

Exercise

Define a Palindrome function that take one word in string as input and return True if it is Palindrome else return False

Palindrome: word that reads same backwards as forwards

example:
is_palindrome ("madam") → True
is_palindrome ("horse") → False

logic:

step 1 : reverse the string
step 2 : compare reversed string with original string.

```
def is_palindrome (word):

    reversed_word = word [::-1]
    if word == reversed_word:
        return True
    else:
        return False

print (is_palindrome ("madam"))
print (is_palindrome ("horse"))
```

output : True
False

Scanned by CamScanner

Date __/__/__

```
def is_palindrome (word):
    reversed_word = word [::-1]
    if word == reversed_word:
        return True
    return False


def is-palindrome (word):
    if word == word [::-1] :
        return True
    return False

def is_palindrome (word):
    return word == word [::-1]
```

Scanned by CamScanner

Fibonacci Series Program

fibonacci Series :
0 1 1 2 3 5 8 13 21 34

1 → 0
2 → 0 1
3 → 0 1 1

```
def fibonacci_seq(n):
    a = 0          # first number
    b = 1          # second number
    if n == 1 :
        print(a)
    elif n == 2 :
        print(a,b) # a b , 0 1
    else:
        print(a,b,end = " ")
        for i in range(n-2):
            c = a+b
            a = b
            b = c
            print(b, end = " ")

print(fibonacci_seq(10))
```

fibonacci_seq(10)
output : 0 1 1 2 3 5 8 13 21 34

Scanned by CamScanner

Date __/__/____

Default Parameters

```
def user_info (first_name, last_name, age):
    print (f" first name = {first_name} ")
    print (f" last name = {last_name} ")
    print (f" age = {age} ")
user_info ('Sangam', 'khare', 18)
```

output : first name = Sangam
last name = khare
age = 18

```
def user_info (first_name, last name, age):
    print (f" first name = {first_name} ")
    print (f" last name = {Last_name} ")
    print (f" age = {age} ")
user_info ('Sangam', 'khare')
```

output :
TypeError : user_info () missing 1 required positional argument: 'age'

Scanned by CamScanner

~~def user_info(first_name,~~

[def user_info(first_name, Last_name, age=

user_info ('Sangam', 'khare')

output : first name = Sangam
last name = khare
age = 23

[def user_info (first_name, Last_name, age=2

user_info ('Sangam', 'khare', 24)

output : first name = Sangam
last name = khare
age = 24
[~~23~~ 24, overwrite]

```
def user_info (first, last = 'unknown',
                age = None):
    print (f" first name = {first} ")
    print (f" Second name = {last} ")
    print (f" age = {age} ")
user_info ('Sangam')
```

Scanned by CamScanner

Date

output : first name = Sangam
Second name = unknown
age = None

```
def user info (first, last='unknown', age):
    user_info ('Sangam', 24)
```

output :
SyntaxError : non-default argument follows
default argument

```
def user_info (first, last='unknown', age =
                                None):
    user_info ('Sangam', 18)
```

output : first name = Sangam
last name = unknown
age = 18

```
def user_info (first='unknown', last,
                                age=None):
    user_info ('Sangam', 24)
```

output :
SyntaxError : non-default argument follows
default argument

Scanned by CamScanner

Date __/__/____ Page No.: ____

```
def user_info (first = 'unknown', last =
               'unknown', age = None):

  user_info ()
```

Output : first name = unknown
Last name = unknown
age = None

Default Parameters: unknown, none

Default Parameters always makes (declares) after non-default Parameters.

Scanned by CamScanner

Date / /

Variable Scope

```
def func():
    x = 7
    return x

def func2():
    print(x)

func2()
```

[Scope of x only in this function. We can't use x outside this function]

output : NameError : name 'x' is not defined

```
def func():
    x = 7 # local variable
    return x

print(x)
```

[Because scope of x only in this function func().]

output : NameError : name 'x' is not defined

local variables : Variables which define inside the function are called as local variables.

```
def func():
    x = 7
    return x

print(func())
```

output : 7

"No real change in history has ever been acheived by discussions." —Subhash chandra bose

Scanned by CamScanner

Date ___/___/___ Page No.:

ex.-

```
x = 5            # global variable
def func():
x = 7            # local variable
return x

print(func())
print(x)
```

output : 7
5

ex.-

```
x = 5            # global variable
def func():
    global x
    x = 7        # local variable
    return x

print(func())
print(x)
```

output : 7
7

ex.-

```
x = 5
def func():
    global x
    x = 7
    return x

print(x)
print(func())
```

Output : 5
7

"To manifest the divinity is the right of everyone" –Sri Ramakrishna

Scanned by CamScanner

Date ___/___/___

ex:-

```
x = 5
def func ():
    global x
    x = 7
    return x

print (x)
print (func ())
print (x)
```

output : 5
7

Scanned by CamScanner

Intro to List

List : List is a collection which is ordered and changeable.
Allows duplicate members.
It is a data structure [ordered collection of items]
you can store anything inside Lists int, float, string.
It can hold any type of data.

ex- numbers = [1, 2, 3, 4] → List
print (numbers)

output : [1, 2, 3, 4]

ex- words = ['word1', 'word2', 'word3']
print (words)

output : ['word1', 'word2', 'word3']

ex- mixed = [1, 2, 3, 4, 'five', 'Six', 2.3, None]
print (mixed)

output : [1, 2, 3, 4, 'five', 'Six', 2.3, None]

Scanned by CamScanner

Date __/__/____

ex- numbers = [1, 2, 3, 4]
print (numbers [1])

output : 2

ex- words = ['word1', 'word2', 'word3']
print (words [:2])

output : ['word1', 'word2']

ex- mixed = [1, 2, 'three', 2.3, None]
print (mixed [-1])

output : None

mixed [1] = 'two'
print (mixed)

output : [1, 'two', 'three', 2.3, None]

mixed [1:] = 'two'
print (mixed)

output : [1, 't', 'w', 'o']

mixed [1:] = ['three', 'four']
print (mixed)

output : [1, 'three', 'four']

Scanned by CamScanner

Add data to List

```
fruits = ['grapes', 'apple']
fruits.append ('mango')
print (fruits)
```

output : ['grapes', 'apple', 'mango']

append() method adds item in list at last position.

ex:-

```
fruits = []
fruits.append ("mango")
fruits.append ("grapes")
print (fruits)
```

output : ['mango', 'grapes']

Scanned by CamScanner

Date __/__/__

More methods to add Data

Insert method

```
fruits1 = ['mango', 'orange']
fruits1.insert(1, 'grapes')
print(fruits1)
```

output : ['mango', 'grapes', 'orange']

ex-
```
fruits1 = ['mango', 'banana']
fruits1.insert(20, 'grapes')
print(fruits1)
```

output : ['mango', 'banana', 'grapes']

ex-
```
fruits1 = ['mango', 'banana']
fruits1.insert(2, 'orange')
print(fruits1)
```

output : ['mango', 'banana', 'orange']

Scanned by CamScanner

how to Join (concatenate) two list

ex:-
```
fruits1 = ['mango', 'orange']
fruits2 = ['grapes', 'apple']
fruits = fruits1 + fruits2
print (fruits)
```

output : ['mango', 'orange', 'grapes', 'apple

Extend method

```
fruits1 = ['mango', 'orange']
fruits2 = ['grapes', 'apple']
fruits1.extend (fruits2)
print (fruits1)
print (fruits2)
```

output: ['mango', 'orange', 'grapes', 'apple']
['grapes', 'apple']

```
fruits1. append (fruits2)
print (fruits1)
print (fruits2)
```

output: ['mango', 'orange', ['grapes', 'apple
['grapes', 'apple']

Scanned by CamScanner

Date / /

Delete data from List

```
fruits = ['orange', 'apple', 'pear', 'banana',
          'kiwi']

#pop method

fruits.pop()
print (fruits)
```

Output : ['orange', 'apple', 'pear', 'banana']

if you dont pass any argument in pop method, it deletes last element of the class. (bydefault)

```
fruits = ['orange', 'apple', 'pear', 'banana', 'kiwi']

fruits.pop (1)
print (fruits)
```

output : ['orange', 'pear', 'banana', 'kiwi']

```
# del operator

del fruits [1]
print (fruits)
```

output : ['orange', 'pear', 'banana', 'kiwi']

Scanned by CamScanner

remove method

ex.- fruits.remove('banana')
print(fruits)

output : ['orange', 'apple', 'pear', 'kiwi']

ex.- fruits.remove('mango')
print(fruits)

output : ValueError : list.remove(x): x not in list

fruits = ['orange', 'apple', 'pear', 'banana', 'apple', 'kiwi']

fruits.remove('apple')
print(fruits)

output : ['orange', 'pear', 'banana', 'apple', 'kiwi']

For add data : append, extend, insert

For delete data : pop, remove, del

Scanned by CamScanner

Date / /

In keyword with list

```
fruits = ['orange', 'apple', 'pear', 'banana',
          'kiwi']

if 'apple' in fruits:
    print ("apple is present")
else:
    print (" not present ")
```

output : apple is present

```
if 'mango' in fruits:
    print ("mango is present")
else:
    print (" not present ")
```

output : not present

Scanned by CamScanner

Some more list methods

```
# count
# sort method
# sorted function
# reverse
# clear
# copy

fruits = ['orange', 'apple', 'pear', 'banana',
          'kiwi', 'apple', 'banana']
# count
print(fruits.count('apple'))
```

output : 2

```
print(fruits.count('orange'))
```

output : 1

```
# sort method

fruits.sort()
print(fruits)
```

output : ['apple', 'apple', 'banana', 'banana',
'kiwi', 'orange', 'pear']

```
numbers = [3, 5, 1, 9, 10]
numbers.sort()
print(numbers)
```

output : [1, 3, 5, 9, 10]

Scanned by CamScanner

sorted function

```
numbers = [3,5,1,9,10]

print(sorted (numbers))
print(numbers)
```

output : [1,3,5,9,10]
[3,5,1,9,10]

clear

```
numbers = [3,5,1,9,10]

numbers.clear()
print (numbers)
```

output : []

copy

```
numbers = [3,5,1,9,10]

numbers_copy = numbers.copy()

print(numbers_copy)
```

Output : [3,5,1,9,10]

Scanned by CamScanner

is vs equals

```
# compare lists
#   ==, is

fruits1 = ['orange', 'apple', 'pear']
fruits2 = ['banana', 'kiwi', 'apple', 'banana']
print (fruits1 == fruits2)
output: False
fruits3 = ['orange', 'apple', 'pear']
print (fruits1 == fruits3) # values are same
output: True
print (fruits1 is fruits3) # values are at same place in memory (both lists)
output: False
```

[fruits1 and fruits3 both are different objects]

Scanned by CamScanner

join and split method

```
# split method
# convert string to list
user_info = 'harshit 24'
user_info.split()
print(user_info)
```

output : ['harshit', '24']

```
user_info = 'harshit,24'
user_info.split(',')
print(user_info)
```

output : ['harshit', '24']

```
name, age = 'harshit,24'.split(',')
print(name)
print(age)
```

output : harshit
24

```
name,age = input("enter
    your name and age").
        split(',')
print(name)
print(age)
```

Scanned by CamScanner

Page No.:

```
#join method
# convert list to string

user_info = ['harshit', '24']

print(',' .join(user_info))
```

output : harshit, 24

ex.-
```
user_info = ['harshit', 24]

print(',' .join(user_info))
```

output : TypeError : sequence item 1 :
expected str instance, int found

Scanned by CamScanner

Date / /

List vs Array

Array: C, C++, Java
int or string or char or float etc.
[fix datatype]

Array is an ordered collection of items.

List: store any data / flexible
int, float, string etc.

In python - array module (import)
[fix datatype]

[numpy module] Performance is better
But we use numpy arrays, not normal array.
Javascript array almost same as
Python List. # flexible

In data science we use numpy arrays.

numpy arrays : binding with C libraries

numpy array's performance is very fast
because C programming language is
very fast.

Scanned by CamScanner

List vs String

strings are immutable

Lists are mutable

immutable : immutable means you can not change your string.

immutable object can not be changed after it is created.

```
s = "string"

t = s.title()

print (t)
print (s)
```

output : String
string

s = "string" s.title() print(s) output : string	s = "string" t = s.title() print (t) output : String

Scanned by CamScanner

mutable : mutable object canbe changed after it is created.

lists are mutable.

```
l = ['word1', 'word2', 'word3']

l.pop()
print(l)
```

output : ['word1', 'word2']

```
l = ['word1', 'word2', 'word3']

l.append ("word3")
print (l)
```

output : ['word1', 'word2', 'word3', 'word3']

In language like ruby, strings are mutable.

Scanned by CamScanner

Looping in List

```
fruits =['orange', 'apple', 'pear', 'banana',
                                 'kiwi']
# for loop

for fruit in fruits:
    print(fruit)
```

Output : orange
apple
pear
banana
kiwi

```
# while loop

i = 0
while i < len(fruits):
    print(fruits[i])
    i += 1
```

Output : orange
apple
pear
banana
kiwi

Scanned by CamScanner

Page No.:

Date / /

List inside List [2d list]

```
matrix = [[1,2,3],[4,5,6],[7,8,9]]
# 3 items → 3 List
print(matrix[0])
output : [1,2,3]
print(matrix[2])
output : [7,8,9]
for i in matrix:
    print(i)

output : [1,2,3]
         [4,5,6]
         [7,8,9]

for sublist in matrix:
    for i in sublist:
        print(i)
output: 1
        2
        3
        4
        5
        6
        7
        8
        9
```


Scanned by CamScanner

Date __/__/____ Page No.:

ex:- print(matrix[1][1])

output : 5

ex:- print(matrix[2][0])

output : 7

```
# type function [type()]
s = "string"
print(type(s))
```

output : <class 'str'>

```
print(type(matrix))
```

output : <class 'list'>

Scanned by CamScanner

More about lists

```
#generate list with range function
numbers = list(range(1,11))
print(numbers)
```

output: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

```
#something more about pop method
numbers = list(range(1,11))
numbers.pop()
print(numbers)
```

output: [1, 2, 3, 4, 5, 6, 7, 8, 9]

```
numbers = list(range(1,11))
print(numbers.pop())
```

output: 10

```
numbers = list(range(1,11))
popped_item = numbers.pop()
print(popped_item)
```

output: 10

Scanned by CamScanner

index method

```
numbers = list(range(1,11))
print(numbers.index(1))
```

output : 0

ex.- numbers = [1,2,3,4,5,6,7,8,9,10,1]

```
print(numbers.index(1))
```

by default searching starts from 0 th position

output : 0

```
print(numbers.index(1,3))
```

→ start searching

output : 10

ex:- numbers = [1,2,3,1,4,5,1]

```
print(numbers.index(1,4))
```

output : 6

x.- numbers = [1,2,3,4,5,6,7,8,9,10,1,5,7,8,1]

start search → stop search

```
print(numbers.index(1,11,14))
```

output : ValueError : 1 is not in list

Scanned by CamScanner

Date __/__/__

Pass list to a function

```
numbers = [1,2,3,4,5,6,7,8,9,10]

def negative_list (L):
    negative = []
    for i in L:
        negative.append (-i)
    return negative

print(negative_list (numbers))
```

output : [-1, -2, -3, -4, -5, -6, -7, -8, -9, -10]

Scanned by CamScanner

Exercise

define a function which will take list containing numbers as input and return list containing square of every elements.

```
numbers = [1, 2, 3, 4]

def square (L):
    sqlist = []
    for i in L:
        sqlist.append (i**2)
    return sqlist

print(square(numbers))
```

output : [1, 4, 9, 16]

```
numbers = list(range(1, 11))
~~print(list(range(1, 11)))~~
print(square(numbers))
```

output : [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]

Scanned by CamScanner

Date / /

Exercise

define a function which will take list as a argument and this function will return a reversed list.

example
[1, 2, 3, 4] → [4, 3, 2, 1]
['word1', 'word2'] → ['word2', 'word1']

Note : you simply do this with reverse method or [::-1]

but try to do this with the help of append and return method.

```
def reverse_list (L):
    return L.reverse ()

numbers = [1, 2, 3, 4]
print (reverse_list (numbers))
```

output : None

```
def reverse_list (L):
    L.reverse ()
    return L

numbers = [1, 2, 3, 4]
print (reverse_list (numbers))
```

Scanned by CamScanner

Date ___/___/_____ Page No.: ____

output : [4, 3, 2, 1]

```
def reverse_list (l):
    return l[::-1]

numbers = [1, 2, 3, 4]

print (reverse_list (numbers))
```

output : [4, 3, 2, 1]

```
def reverse_list (l):
    r_list = []
    for i in range (len (l)):
                    [for i in range
                        (1, len(l)+1):
        popped_item = l.pop()
        r_list.append (popped_item)
    return r_list

numbers = [1, 2, 3, 4]

print (reverse_list (numbers))
```

output : [4, 3, 2, 1]

Scanned by CamScanner

Date / /

Exercise

define a function that take list of words as argument and return list with reverse of every element in that list.

```
# example
# ['abc', 'tuv', 'xyz'] -> ['cba', 'vut', 'zyx']

word = ['abc', 'tuv', 'xyz']

def reverse_elements (l):
    elements = []
    for i in l:
        elements.append (i[::-1])
    return elements

print(reverse_elements (words))
```

output : ['cba', 'vut', 'zyx']

Scanned by CamScanner

Date / /

Exercise

Page No.:

```
# filter odd even
# define a function
# input
# list --> [1,2,3,4,5,6,7]

# output --> [[1,3,5,7], [2,4,6]]

def filter_odd_even(l):
    odd_nums = []
    even_nums = []
    for i in l:
        if i%2 == 0:
            even_nums.append(i)
        else:
            odd_nums.append(i)
    returning = [odd_nums, even_nums]
    return returning

nums = [1,2,3,4,5,6,7]
print(filter_odd_even(nums))
```

output : [[1,3,5,7],[2,4,6]]

Scanned by CamScanner

Date / /

Exercise

Common elements finder function
define a function which takes 2 lists as input and return a list which contains common elements of both lists.

example
input → [1,2,5,8] , [1,2,7,6]

output → [1,2]

```
def common_finder (l1, l2):
    output = []
    for i in l1:
        if i in l2:
            output.append (i)
    return output

print (common_finder([1,2,5,8], [1,2,7,6]))
```

output: [1,2]

Scanned by CamScanner

min and max functions

```
numbers = [6, 60, 2]
print(min(numbers))
```

output : 2

```
print(max(numbers))
```

output : 60

ex:-

```
def greatest_diff(l):
    return max(l) - min(l)
print(greatest_diff(numbers))
```

output : 58

Scanned by CamScanner

Date

Exercise

define function
[1, 2, 3, [1, 2], [3, 4]] - input

output : 2 (no. of list inside list)

type() function

```
def sublist_counter (l):
    count = 0
    for i in l:
        if type(i) == list:
            count += 1
    return count

mixed = [1, 2, 3, [1, 2], [3, 4]]

print (sublist_counter (mixed))
```

output : 2

Scanned by CamScanner

Tuples

Intro to Tuples

Tuple : A Tuple is a collection of Python objects separated by commas. In someway a tuple is similar to a list in terms of indexing, nested objects and repetition but a tuple is immutable unlike lists which are mutable.

Python tuples are written with round brackets. You can't change values (add or del) from tuple once it created.

tuple data structure
tuple can store any data type

most important tuples are immutable, once tuple is created you can't update.
data inside tuple

example = ('one', 'two', 'three')

no append, no insert, no pop, no remove

days = ('monday', 'tuesday', 'wednesday', 'thursday', 'friday', 'saturday', 'sunday')

like this type of data when our data will not change.

We use tuples because tuples are faster than lists (performance).

Scanned by CamScanner

Date __/__/__

Use methods in tuple
count , index
len function
slicing

```
print (example [:2])
```

output : ('one' , 'two')

```
example [0] = 1
print (example)
```

output : TypeError : 'tuple' object does not support item assignment

Scanned by CamScanner

More about tuples

looping in tuples

```
mixed = (1, 2, 3, 4.0)

for i in mixed :
    print (i)
```

output : 1
2
3
4.0

note: You can use while loop too

tuple with one element

```
nums = (1)
words = ('word1')
```

These are not tuples

```
print (type (mixed))

print (type (nums))
print (type (words))
```

output : <class 'tuple'>
<class 'int'>
<class 'str'>

tuple with one element : -

Scanned by CamScanner

Page No.:

Date __/__/____

```
nums = (1,)
words = ('word1',)

print (type (nums))
print (type (words))

output : <class 'tuple'>
         <class 'tuple'>

# tuple without parenthesis

guitars = 'yamaha', 'baton rouge', 'taylor'

print (type (guitars))

output : <class 'tuple'>

# tuple unpacking

guitarists = ('Maneli Jamal', 'Eddie Van Der
              Meer', 'Andrew Foy')

guitarist1, guitarist2, guitarist3 = (guitarists)

print (guitarist1)

output : Maneli Jamal

print (guitarist2)
output: Eddie Van Der Meer

print (guitarist3)
output : Andrew Foy
```

"No real change in history has ever been achieved by discussions." –Subhash chandra bose

Scanned by CamScanner

Date ___/___/___

```
guitarist1, guitarist2 = (guitarists)
print(guitarist1)
```

output : ValueError : too many values to unpack (expected 2)

```
# list inside tuples
favoriates = ('southern magnolia', ['Tokyo Ghoul Theme', 'landscape'])

favoriates[1].pop()

print (favoriates)
```

output : ('southern magnolia', ['Tokyo Ghoul Theme'])

```
favoriates[1].append("we made it")

print (favoriates)
```

output : ('southern magnolia', ['Tokyo Ghoul Theme', 'we made it'])

```
# some functions that you can use
  with tuples.
# min(), max(), sum()
print (min(mixed))
```

output : 1

Scanned by CamScanner

Date __/__/____

```
print(max(mixed))
```

output : 4.0

```
print(sum(mixed))
```

output : 10.0

Scanned by CamScanner

Function returning two values

```
def func (int1, int2):
    add = int1 + int2
    multiply = int1 * int2
    return add, multiply

print (func (2, 3))
```

output : (5,6) # tuple

```
add, multiply = func(2,3)
print (add)
print (multiply)
```

output : 5
6

Something more about tuples, list, str

```
nums = tuple (range (1,11))

print (nums)
```

output : (1,2,3,4,5,6,7,8,9,10)

```
nums = list ((1,2,3,4,5,6,7,8,9,10))

print (nums)
```

output : [1,2,3,4,5,6,7,8,9,10]

Scanned by CamScanner

```
nums = str((1,2,3,4,5,6,7,8,9,10))

print(nums)
```

String

```
output : (1,2,3,4,5,6,7,8,9,10)

print(type(nums))

output : <class 'str'>
```

["(1,2,3,4,5,6,7,8,9,10)"]

```
nums_list = str([1,2,3,4,5,6,7,8,9,10])

print(nums_list)
print(type(nums_list))

output : [1,2,3,4,5,6,7,8,9,10]
         <class 'str'>
```

Scanned by CamScanner