Keywords / system defined words / Reserved word
- 35

- True, False, None, is, in, not, and, or, import, from, for, while, del, yield

- Kwlist — 35 keywords
  ↓
  Keyword module.

Keyword — word with spcl meaning in computer

29/7/25

# ++ and -- operator:

a = 25

print (++a) — # +(+a) — +a ⇒ +25

print (a++) — # (a+)+ — a+ ⇒ 25+ Error

print (a++1) — # (a+)(+1) — a+1 ⇒ 25+1 ⇒ 26

print (--a) — # -(-a) — +a ⇒ +25

print (a--) — # (a-)- — a+ ⇒ 25+ ⇒ Error

print (a--1) — # (a-)(-1) — a+1 ⇒ 25+1 ⇒ 26

print (-a) — # -25

print (+-a) — # +(-a) -- a ⇒ -25

print(-+a) — # -(+a) — (-a) ⇒ -25

# floor() & ceil()

import math

print (math.floor(10.8)) — # 10

print (math.ceil(10.4)) — # 11

print (math.floor(25.0)) — # 25

print (math.ceil(25.0)) — # 25

pit (math.floator(-3.5) — # -4

print (math.ceil(-3.5) — # -3

```
print(math.floor(-9.0))  — # -9
print(math.ceil(-9.0))  — # -9
print(math.floor(25.1))  — # 25
print(math.ceil(25.1))  — # 26
print(floor(3.5))  — # 3
print(ceil(3.5))  — # 4
```

# gcd():

```
print(math.gcd(12,15))  — # 3
print(math.gcd(12,18))  — # 6
print(math.gcd(4,7))  — # 1
print(math.gcd(7,7))  — # 7
print(math.gcd(-8,-27))  — # Error +1
print(math.gcd(-4,6))  — # Error +2
print(math.gcd(0,7))  — # 7
print(math.gcd(3,0))  — # 3
print(math.gcd(0,0))  — # 0
print(math.gcd(15,15))  — # 5
```

# abs():-

```
from builtins import abs
print(abs(-35.8))  — # 35.8
print(abs(-27))  — # 27
print(abs(29.5))  — # 29.5
print(abs(32))  — # 32
import builtins
print(builtins.abs(-25))  — # 25
```

# max(), min():

```
from builtins import max, min
print (max (10.8 , 20.6))   —— # 20.6
print (min (10.8, 20.6, 5.9, 12.3))   —— #5.9
print(max(25, 10.8))   —   #  25
import builtins
print (builtins. max (10, 20, 30))   —  # 30
print (builtins. min (10, 20, 15, 5, 12)) # 5
```

# pow():

```
from builtins import pow
print (pow (10, -2))   —— #  10^-2
print (pow (4, pow (3, 2)))   —  #  4 pow(3^2)
print import builtins
print (builtins. power (2, 3))   —  #  2^3 — 8
print (builtins. power (-2, -3))   -  2^-3
```

# Find outputs:

How to import kw list — # import kwlist ~~from~~ keyword import *

How to print kw list — # print (keyword. kwlist)

How to print no. of keywords — print len(kwlist)

How to print type of kwlist — print (type (keyword. kwlist

print (keyword. kwlist ——

# Find oiltpts:

How to import keyword module — # import keyword.

How to ~~import~~ print kw list — # print (keyword. kwlist)

How to print number of keywords - print(len(keyword.kwlist))

How to print type of kwlist - print(type(keyword.kwlist))

print(kwlist) -# prints all keywords.

```
from mod1 import *
from mod2 import *
print(x)
print(y)
```

```
import mod1, mod2
print(mod1.x)
print(mod2.y)
```

---

```
# input()
   x = input('Enter input :') # Rama Roo
                  ↳ always reads as string
    print(type(x))
    print(x)
# x = int(input('Enter input'))
    - float(input("Enter input")
```

# eval():

```
print(eval('25')) — # 25
print(eval('10.8')) — # 10.8
print(eval('False')) — # False
print(eval('3+4j')) — # 3+4j
print(eval('Hyd')) — # Error
print(eval(" 'Hyd' ")) — 'Hyd'
print(eval("3+4*5")) — # 23
print(eval('[10,20,15,18]')) — [10,20,15,18]
print(eval('{10,20,15,18,20,12,15}')) — {10,20,15,18,12} No duplicates
                  String
print(eval('(10,20,30)')) — # (10,20,30)
print(eval("{10:'Hyd', 10:'Sec'}")) — {10:'Sec'}
print(eval(4+5)) — # Error
          should be string
```

# Find output:

```
print (eval(" hyd ")) ——— # hyd
hyd = 'sec' ——— # sec is assigned
print (eval ('hyd')) ——— # sec
sec = '25' ——— # 25
print (eval('sec')) ——— # 25
print(eval(sec)) ——— # 10.8 as assigned
cyb = 10.8
print (eval ('cyb')) ——— # 10.8
print(eval (cyb)) ——— # Error.
```

# Find output:

```
print (bool('False')) ——— # True
print (eval ('False')) ——— # False.
print (bool('')) ——— # False
print (eval(' ')) ———
print (eval('')) ——— # empty string
print (eval (' ')) 
print (eval (' '))
```

# Sep argument

```
a, b, c = 25, 10.8, 'Hyd'
print (a,b,c, sep = ';') ——— # 25;10.8;Hyd.
print (a,b,c, sep = '\t') ——— # 25 <tab> 10.8 tab Hyd <tab>
print (a,b,c, sep = '---') ——— # 25---10.8---Hyd
print (a,b,c) ——— # 25 10.8 Hyd.
print (a,b,c, separate = ':') # 25 : 10.8 : Hyd
                  Error.
```

# First output:

```
a,b,c = 25, 10.8, 'Hyd'
print(a,b,c, end='---')  — # 25 10.8 Hyd---
print(a,b,c, sep='...')  — # 25...10.8...Hyd
print(a,b,c, sep=':::', end='\t\t\t')  25:::10.8:::Hyd:::<tab><tab><tab>
print(a,b,c) # 25 10.8 Hyd
```

# First output:

```
print('Hyd')   — # Hyd
print()        — # Returns Nothing printed
print('Sec')   — # Sec
print()        — # returns Nothing printed
print('cyb')   — # cyb
```

# Find outputs:

```
l = [10,20,30,40]
t = (10,20,30,40)
s = {10,20,30,40}
print(l,t,s)   —— # [10,20,30,40] <space>
                    (10,20,30,40) <space>
                    {10,20,30,40} <space>
```

# Find outpts:

```
a = 25
b = '%.f' % a   — # 25.000000
    format  obj
print(b)
print(type(b))  — # <class 'str'>
x = 10.8        — # 10
y = '%d' % x
print(y)        — # <class 'str'>
print(type(y))  — [10,20,15,11]
m = [10,20,15,15]
n = '%s' % m
```

# Find output

```
a = 10.9274
print ('%8.2f' %a)
```

%d or %i — int
%g or %f — float
%s — string

<3 spaces > 10.93

```
print ('%9.1f' %a)    — # <1space> 10.92
print ('%10.3f' %a)   — #
print ('%of' %a)      — <8space>
print ('%6f', %a)     — <4space>
print ('%f' %a)       —
```

# Find outputs

```
a = [10,20,30,40]     — # [10,20,30,40]
print ('%s' %a)       — # [10,20,30,40]
print ('%s' a)        — # Syntax Error
print ('%s', %a)      — # Syntax Error
print ('%l' %a)       — # Syntax Error
print (a)             — # [10,20,30,40]
```

---

# f string

format (f)(f) - Conversion of any python object to 'string'

① F'{object}' — value of obj — F'{a}\t{b}\t{c}
(Syntax)

② f'{object=}' — objectname = value — F'{a=}\t{b=}\t{c=}

① Program :

```
import math
a = int (input ("Enter the first number : "))
b = int (input ("Enter the second number : "))
sum_ab = a+b
diff_ab = a-b
prod_ab = a*b
quotient_ab = a/b if b!=0 else m/s → 0/0 3/0
remainder_ab = a%b if b!=0 else undefined
largest = max (a,b)
smallest = min (a,b)
sqrt_a = math.sqrt(a) if a>0=0 else undefined
power_ab = a**b
gcd_ab = math.gcd (a,b)
print (f"Sum of two numbers : {summation}")
print (f" Diff of two numbers : {difference sum ab}")
print (f" Product of two number : {prod_ab}")
print (" f" Quotient of two number : {ab quotient_ab}")
print (f" Remainder of two number : {remainder_ab}")
print (f" Largest of two number : {largest}")
print (f" smallest of two number : {smallest}")
print (f" power of two numbers : {power_ab}")
print (f"gcd of two number : {gcd_ab}")

(F"sqrt({a}) = {math.sqrt({a})}
```

② Swapping

```
x = 25
y = 'Hyd'
x, y = y, x
print(f" x = {x} , y = {y} ")
```

After printing: x = Hyd, y = 25

③ Largest among 3

```
a = int(input("Enter 1st number : "))
b = int(input("Enter 2nd number : "))
c = int(input("Enter 3rd number : "))
```

100 → 25 (x), 10 (y), 25 (z)

a b c
4 > 5 6
4 > 6 — false
4 > 5 — false

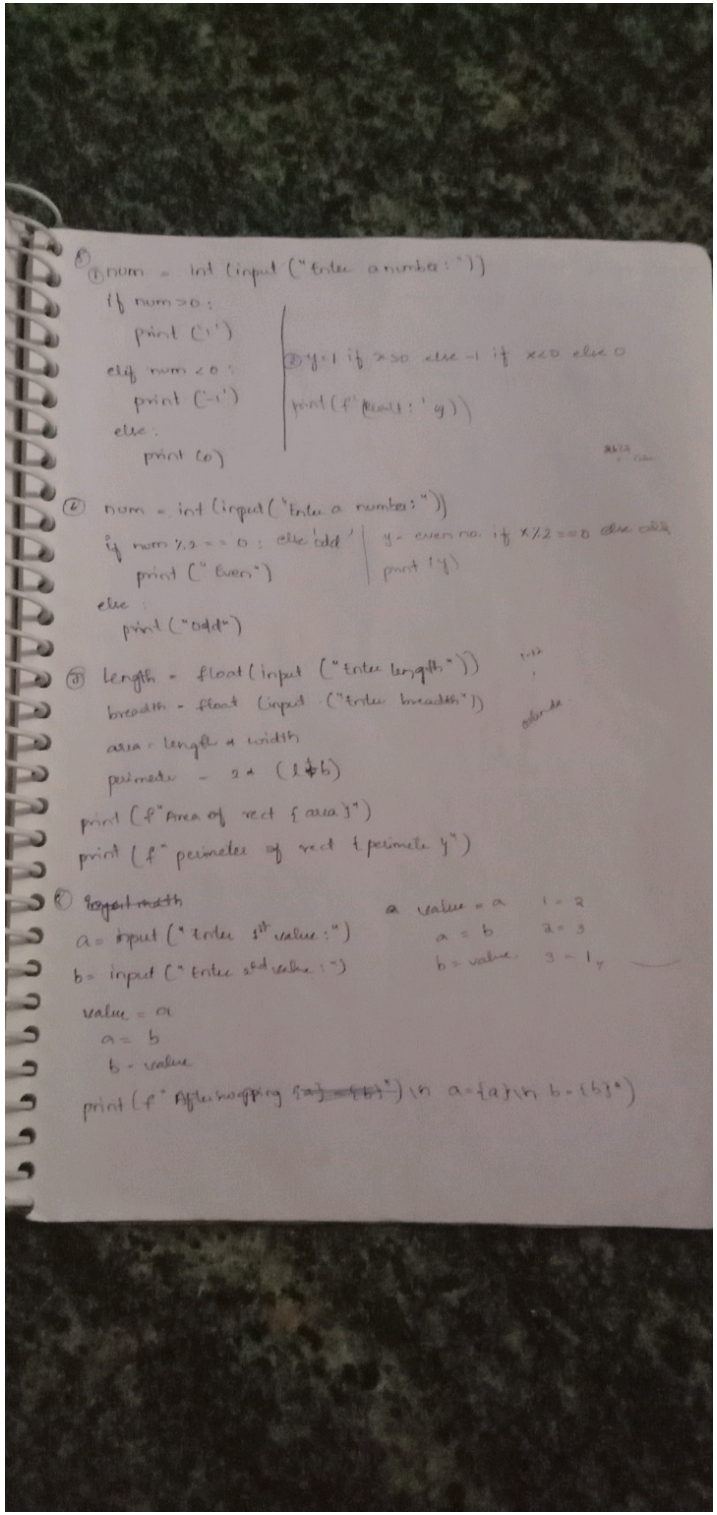
```
if a>b and a>c :
    largest = a
elif b>=a and b>=c :
    largest = b
else :
    largest = c
print(f" largest of a,b,c is : {largest} ")
```

```
max = a if a>b and a>c else b if b>c else c
print(f'largest:{max}')
```

④ <, >, =

```
a = int(input("Enter 1st number : "))
b = int(input("Enter 2nd number : "))
if a>b :
    print('>')  # A is > than b
elif a<b :
    print('<')  # A < than b
else :
    print('=')  # A = b
```

c = '>' if a>b else '<' if a<b else '='



① num = int (input ("Enter a number: "))
if num > 0:
    print ('1')
elif num < 0:
    print ('-1')
else:
    print (0)

② y = 1 if x > 0 else -1 if x < 0 else 0
print (f"Result: {y}")

② num = int (input ("Enter a number: "))
if num % 2 == 0: else 'odd'
    print ("Even")
else:
    print ("Odd")

y = even no. if x % 2 == 0 else odd
print (y)

③ length = float (input ("Enter length"))
breadth = float (input ("Enter breadth"))
area = length * width
perimeter = 2 * (l + b)
print (f"Area of rect {area}")
print (f"perimeter of rect {perimeter}")

④ ~~import math~~
a = input ("Enter 1st value:")
b = input ("Enter 2nd value:")
value = a
a = b
b = value
print (f"After swapping a={a} \n b={b}")

a value = a  1 = 2
a = b  2 = 3
b = value  3 = 1

(9)
```
a = input("Enter first value:")
b = input("Enter second value:")
a, b = b, a
print(f"After swapping: a = {a}, b = {b}")
```