Python Keywords Reference Table

Keyword	Meaning / Function	Example Code
and	Logical AND operator (returns True if both are True)	if a > 0 and b > 0:
as	Used to create an alias (commonly in imports or exceptions)	import math as m
assert	Used for debugging; checks if a condition is True	assert x > 0, "x must be positive"

break	Exits a loop early	for i in range(5):
		  if i==3:
		  break

class	Defines a class	class Car: pass

continue	Skips current loop iteration	for i in range(5):
		  if i==2:
		  continue

def	Defines a function	def greet(): print('Hi')
del	Deletes an object or variable	del x

elif	Else if condition in control flow	if x>0:
		  ...
		elif x==0:

else	Executes if condition is False	if x>0:
		  ...
		else:

except	Handles exceptions	try:
		  1/0
		except ZeroDivisionError:
		  print('Error')

False	Boolean value for false	flag = False

finally	Runs code after try/except, no matter what	try:
		  ...
		finally:
		  print('Done')

for	Loop through sequence	for i in range(5): print(i)
from	Used to import specific parts of a module	from math import sqrt
global	Declares a global variable inside a function	global x
if	Conditional statement	if x > 10:
import	Imports a module	import math
in	Checks membership in a sequence	if 3 in [1,2,3]: print('Yes')
is	Tests object identity (same object in memory)	if a is b:
lambda	Creates an anonymous function	square = lambda x: x*x
None	Represents no value	x = None

Keyword	Meaning / Function	Example Code
nonlocal	Refers to variable in outer (non-global) function scope	nonlocal count
not	Logical NOT operator	if not done:
or	Logical OR operator	if a==1 or b==1:
pass	Does nothing (placeholder)	def func(): pass
raise	Raises an exception	raise ValueError('Invalid input')
return	Returns a value from a function	return x + y
True	Boolean value for true	flag = True
try	Defines a block to test for errors	try: ... except:
while	Repeats code while condition is True	while x < 10:
with	Simplifies resource management (e.g. file handling)	with open('file.txt') as f:
yield	Returns a generator value	yield x
async	Declares asynchronous function	async def run():
await	Waits for async result	await task