

# Python PCAP 31-03 Exam Practice Questions List (Difficult or Reminder)

by [naufalnashif](#)

Courses:

- [Cisco Python Essential 1](#)
- [Cisco Python Essential 2](#)
- [Udemy PCAP 31-03](#)

Exam Practice:

- [Exam Practice Without Answer](#)
- [Exam Practice With Answer](#)

Note :

Passing grade 70/100 dari 40 soal, atau 28/40 soal. Berarti maksimal salah 12 soal.

## Modules & Packages

Soal	Keterangan
Select all valid option(s) about the result of dir() <input type="checkbox"/> A list of filenames inside the directory <input checked="" type="checkbox"/> A list of the module's attribute <input type="checkbox"/> A list of names of class attributes <input type="checkbox"/> A list of names of object attributes <input type="checkbox"/> A list of names of the base class attributes <a href="#">Answer &gt;&gt;&gt;</a>	
math.ceil() or math.floor()	return int
math.factorial(x)	x (int) >= 0 0! = 1
random.choices(populasi, weights, k)	return list
random.sample(populasi, k)	return list

## Soal

## Keterangan

```
1 import platform  
2  
3 print(platform.platform())  
4 print(platform.platform(alienated = 0, terse = 0))  
5 print(platform.platform(alienated = 1, terse = 1))  
6  
7 print(platform.machine())  
8 print(platform.processor())  
9 print(platform.system())  
10 print(platform.version())  
11 print(type(platform.version()))
```

```
→ Linux-6.1.85+-x86_64-with-glibc2.35  
Linux-6.1.85+-x86_64-with-glibc2.35  
Linux-6.1.85+-x86_64-with-glibc2.35  
x86_64  
x86_64  
Linux  
#1 SMP PREEMPT_DYNAMIC Thu Jun 27 21:05:47 UTC 2024  
<class 'str'>
```

The digraph written as `#!` is used to:

- tell an MS Windows OS how to execute the contents of a Python file
- tell a Unix or Unix-like OS how to execute the contents of a Python file
- make a particular module entity private
- create a docstring

shebang or hashbang

memberi tahu path to interpreter untuk dijalankan di Unix OS

### Question 5

When a module is imported, its contents:

- may be executed (explicitly)
  - are executed as many times as they are imported
  - are ignored
  - are executed once (implicitly)
- When a module in Python is imported, **the entire file is executed once implicitly** in order to assure no errors are present in the module.

ketika import module, maka hanya di eksekusi sekali walaupun kita melakukan import berulang ulang

The `sys.stderr` stream is normally associated with:

- the console/terminal
- the screen

## Error Handling (*Exceptions*)

## Soal

## Keterangan

What is the output of the following code?

```
class E (Exception):
    def __init__(self, message):
        self.message = message
    def __str__(self):
        return "Surprise"

try:
    raise Exception("Stop")
except E as e:
    print(e)
else:
    print("Goodbye")

```

Unhandled Exception  
() Surprise  
() Stop  
() Goodbye

What is the output of the following code?

```
try:
    raise Exception
except: Default Except must be last
    print("Spam", end="")
except BaseException:
    print("Ham", end="")
except Exception:
    print("Eggs")
```

```
class SpamException(Exception):
    def __init__(self, message):
        <<< INSERT CODE HERE >>>
        self.message = message
raise SpamException("Spam")
```

[X] super().\_\_init\_\_(message)  
[ ] Exception.\_\_init\_\_(self, message)  
[X] super(SpamException, self).\_\_init\_\_(message)  
[ ] super.\_\_init\_\_(message)

What is the result of the following code?  
=> assert (False, 'Trigger Assertion')  
[ ] No output  
[ ] Trigger Assertion  
[ ] SyntaxError: invalid syntax  
[ ] Assertion is always true

tuple yang memiliki isi dianggap True

assert False, 'Message' akan selalu False

Which of the following are examples of built-in concrete Python exceptions? (Select two answers)

- BaseException
- ArithmeticError
- ImportError
- IndexError

build in konkret ( error plaing dasar/spesifik )

```
1 try:
2     raise Exception
3 except BaseException:
4     print("a")
5 except Exception:
6     print("b")
7 except:
8     print("c")
```

a

BaseException merupakan exception paling luas

# Strings

Soal	Keterangan
<p>What is the output of the following code?</p> <pre>print("C\Program Files\Microsoft\Windows NT", end="") print("\")</pre> <ul style="list-style-type: none"> <li>( ) Syntax Error</li> <li>( ) C:\Program Files\Microsoft\Windows NT</li> <li>( ) Replace escaped characters with "" e.g. C:\program Files\microsoft\indows NT?</li> <li>( ) Ignore escaped characters e.g. C: rogram Filesicrosoftindows NT</li> </ul>	<p>Answer :</p> <ul style="list-style-type: none"> <li>- Syntax Error</li> </ul>
<p>* ASCII , UNICODE, UTF-8, codepoints, escape sequences Select all valid option(s) below about string</p> <ul style="list-style-type: none"> <li>✓ string.ascii_letters is a concatenation of ascii_lowercase and ascii_uppercase</li> <li>[ ] string.ascii_letters is a concatenation of ascii_lowercase, ascii_uppercase and digits</li> <li>[ ] string.ascii_letters are all printable characters found in the keyboard</li> <li>✗ string.ascii_lowercase contains 'abcdefghijklmnpqrstuvwxyz'</li> <li>✗ string.ascii_uppercase contains 'ABCDEFGHIJKLMNPQRSTUVWXYZ'</li> </ul>	<p>Strings : upper + lower  bukan digit dan any char</p>
<p>What is the output of the following code?</p> <pre>spam = chr('a') ham = ord(spam) print(spam, ham)</pre> <ul style="list-style-type: none"> <li>( ) 97 a</li> <li>( ) TypeError: an integer is required (got type str)</li> <li>( ) TypeError: chr() takes exactly two arguments (1 given)</li> <li>( ) TypeError: ord() takes exactly two arguments (1 given)</li> <li>( ) Syntax Error</li> </ul>	<p>Answer : b chr(int) not chr('a')</p>
<p>spam = 'FuBar' ham = spam[:]  ✗ spam == ham ✗ id(spam) == id(ham) ✗ spam.startswith(ham) ✗ spam.endswith(ham) [ ] spam.equals(ham)</p>	<p>Jika <i>immutable</i>, python ga akan membuat memori baru id(spam) == id(ham)</p>
<p>What is the output of the following code?</p> <pre>&gt;&gt;&gt; None * 2</pre> <ul style="list-style-type: none"> <li>( ) 0</li> <li>( ) None</li> <li>( ) NoneNone</li> <li>✗ TypeError: unsupported operand type(s) for *</li> </ul> <p>Answer &gt;&gt;&gt;</p>	<p>not support NoneType</p>
<p>What is the output of the following code?</p> <pre>&gt;&gt;&gt; sorted([5, "1", 100, "34"])</pre> <ul style="list-style-type: none"> <li>( ) [1, 5, "34", 100]</li> <li>( ) [5, "1", "34", 100]</li> <li>( ) ["1", "100", "34", "5"]</li> <li>( ) [1, 5, 34, 100]</li> <li>✗ TypeError: &lt; not supported between instances of 'str' and 'int'</li> </ul>	<p>cannot sort int and str</p>
<p>What is the output of the following code?</p> <pre>spam.txt spam ham eggs spam.py f = open('spam.txt', 'r') if 'eggs' in f:     print('Eggs found') else:     print('Eggs not found')</pre> <ul style="list-style-type: none"> <li>( ) Eggs found</li> <li>✗ Eggs not found</li> <li>( ) TypeError: argument type TextIOWrapper not iterable</li> <li>( ) SyntaxError: invalid syntax</li> </ul>	<p>open(file, 'r') return file object not file content  content = f.read()</p>
<p>Which of the calls below are valid String function calls and will return True?</p> <ul style="list-style-type: none"> <li>✗ 'abc123'.isalnum()</li> <li>✗ 'abc'.isalpha()</li> <li>[ ] '123abc'.isidentifier()</li> <li>✗ '123abc'.islower()</li> <li>✗ '123'.isdigit()</li> <li>✗ 'Abc'.istitle()</li> </ul>	<p>isalnum() -&gt; isalpha() + isnumeric() isnumeric -&gt; termasuk isdigit(0-9)</p>
<p>What is the output of the following code?</p> <pre>&gt;&gt;&gt; sorted(['banana', 'pear', 'grapes', 'apple'], key= lambda x: x[::-1])</pre>	<p>isidentifier() -&gt; cek apakah valid sebagai variable  sorted(iter, key) key = fungsi key = lambda x : x -1 -&gt; reverse setiap element</p>

Soal	Keterangan
<pre>* .sort(), sorted(), .index(), .find(), .rfind() What is the output of the following code? &gt;&gt;&gt; "Spam Ham Eggs" .index( 'Spam' , 1 ) ()</pre> <p>( ) Spam    ( ) 0    ( ) 1    ( ) ValueError: substring not found    ( ) TypeError: index() takes 1 argument (2 given)</p>	<pre>.index(str, start, end) 'Spam' ada di index 0 bukan 1 jadi ValueError = not found</pre>
<pre>* .sort(), sorted(), .index(), .find(), .rfind() What is the output of the following code? t = "Spam Ham" print(t.find( "Ham" , 0 )== t.index( "Ham" , 0 )) print(t.find( "Eggs" , 0 )== t.index( "Eggs" , 0 )) ()</pre> <p>( ) True True    ( ) True False    ( ) False will be printed followed by ValueError: substring not found    ( ) True will be printed followed by TypeError: find() takes 1 argument (2 given)</p> <p><a href="#">Answer &gt;&gt;&gt;</a></p> <pre>* .sort(), sorted(), .index(), .find(), .rfind() What is the output of the following code? t = "Spam Ham" print(t.rfind( "am" ) == t.find( "am" )) print(t.rfind( "am" , 3 ) == t.find( "am" , 3 )) print(t.rfind( "am" , -3 )== t.find( "am" , -3 )) ()</pre> <p>( ) False False False    ( ) False True True    ( ) True True True    ( ) True will be printed followed by TypeError: rfind takes 1 argument (2 given)</p> <p><a href="#">Answer &gt;&gt;&gt;</a></p>	<pre>.index() -&gt; jika tidak ditemukan maka ValueError .find() &amp; .rfind() -&gt; return -1</pre>
<p>UTF-8 is:</p> <p><input type="radio"/> a synonym for byte</p> <p><input checked="" type="radio"/> a form of encoding Unicode code points</p> <p><input type="radio"/> the 9th version of the UTF standard</p> <p><input type="radio"/> a Python version name</p>	<p>UTF – 8 is a form of encoding Unicode code points</p>
<pre>1 print('Anre' &gt; "Anrea") 2 3</pre> <p><b>False</b></p> <pre>1 print('2Anre' &gt; "1Anrea") 2 3</pre> <p><b>True</b></p>	<p>pada string, yang terpanjang yang lebih besar    namun jika ada angka, akan di bandingkan angka paling awal</p>
<p>What will be printed in the following code?</p> <pre>spam = "!!!!!!" ham = """" """ print(spam, ham)</pre> <p>( ) Syntax Error    ( ) Two empty strings    ( ) An empty string and a new line character    ( ) Two new line character</p> <p><a href="#">Answer &gt;&gt;&gt;</a></p>	

## Soal

## Keterangan

s = 'Python'

- ( ) print(s[0] + s[-1])
- ( ) print(s[:5])
- print(s[::-5])
- ( ) print(s[::-1][::-5])

Answer >>

step -5 berarti membaca dari index terakhir dengan step 5 -> nP

sedangkan step 5 -> Pn

1 '' in 'AnyString'

True

string kosong itu termasuk kedalam string apapun

```
1 with open('file.txt', 'w') as f:  
2     f.write('spam ham eggs')  
3  
4 with open('file.txt', 'r') as f:  
5  
6     if 'eggs' in f:  
7         print('eggs found')  
8     else:  
9         print('eggs not found')  
10  
11    print(f.read(), 'tidak ada file karena pointer di akhir')  
12    f.seek(0)  
13    print(f.read())  
14    print(f)
```

eggs not found  
tidak ada file karena pointer di akhir  
spam ham eggs  
<`_io.TextIOWrapper` name='file.txt' mode='r' encoding='UTF-8'>

open() as f ini mengembalikan object bukan content , jadi tidak bisa diiterasi

untuk melakukan iterasi lakukan read dahulu

What is the output of the following code?

```
>>> "/" .join({ "Month" : "12" , "Day" : "25" , "Year" : "2021" })  
( ) 12/25/2021  
→ Month/Day/Year  
( ) Month/12/Day/25/Year/2021  
( ) TypeError: can only join an iterable
```

Answer >>

.join pada dict hanya akan mengambil key

What is the output of the following code?

```
>>> "XYZ" .join( "123" )  
( ) XYZ123  
( ) 123XYZ  
→ 1XYZ2XYZ3  
( ) X123Y123Z  
( ) TypeError: can only join an iterable
```

1 '/spam/ham/eggs/'.split('/')

[' ', 'spam', 'ham', 'eggs', ' ']

menghasilkan string kosong dan panjang list sebanyak /

## OOP

## Soal

## Keterangan

What is the output of the following code?

```
class Foo :  
    bar = 'spam'  
  
f1 = Foo()  
f2 = Foo()  
f2.bar = 'ham'  
Foo.bar = 'eggs'  
print(f1.bar, f2.bar, Foo.bar)  
( ) spam ham eggs  
X eggs ham eggs  
( ) eggs eggs eggs
```

Jika instance tidak melakukan rewrite bar maka bar akan mengambil attr kelas

issubclass(classA, classA)  
will return True

isinheritance(classA, classA)  
will return True

```
1 str_one = 'hello'  
2 str_two = 'hell'  
3 str_two += 'o'  
4  
5 print(str_one is str_two)
```

because when modify string, python will make a different reff memory

is used to compare 2 object memory

False

```
1 class Void:  
2     pass  
3  
4     a = Void()  
5     b = Void()  
6  
7     a is b
```

dua instance pasti memiliki beda referensi memori

False

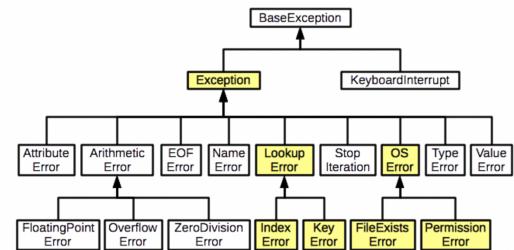
```
try:  
  
except Exception:  
  
except BaseException:  
  
except:
```

sebaiknya dalam membuat urutan exception dari exception yang paling spesifik baru ke yang paling luas

dan untuk except harus di akhir

## Soal

## Keterangan



```
1 try :  
2     raise Exception ('Your Message!')  
3 except Exception as e :  
4     print(e)  
5     print(e.args)
```

↳ Your Message!  
('Your Message!',)

e.args -> tuple

```
1 for subclass in Exception.__subclasses__():  
2     print(subclass.__name__)
```

TypeError  
StopAsyncIteration  
StopIteration  
ImportError  
 OSError  
EOFError  
RuntimeError  
NameError

menghasilkan hierarchy error.

Cara buat exception hanya melakukan inheritance dari error paling bawah

```
1 class MyEx(Exception):  
2     def __init__(self, msg):  
3         Exception.__init__(self, msg+msg)  
4         self.args = (msg,)  
5  
6     try:  
7         raise MyEx('wrong!')  
8     except MyEx as e:  
9         print(e)
```

wrong!

self.args akan mengoverride msg di instance

```
1 try:  
2     val = int(input('Provide a number: '))  
3     print(val/val)  
4 except TypeError:  
5     print('a')  
6 except ValueError:  
7     print('b')  
8 except ZeroDivisionError:  
9     print('c')  
10 except:  
11     print('d')
```

Provide a number: a  
b

type sudah benar str hanya value yang salah

JANGAN GUNAKAN assert dalam production

dengan perintah -O ketika run program di CLI maka semua assert akan unactive

```
sum = lambda x , y : x + y  
sum(10, 1)
```

lambda bisa di assign ke variable

## Soal

```
def count_occurrences(file_name, word):
    word_counts = {}

    try :
        with open(file_name, 'r') as f:
            words = f.read().replace(',', '').replace('.', '').split()
        for w in words:
            w = w.lower()
            if w in word_counts:
                word_counts[w] += 1
            else:
                word_counts[w] = 1
    except:
        print('Something error!')
        raise

    return word_counts[word]
    return word_counts.get(word.lower(), 0)
```

## Keterangan

pendekatan terbaik untuk return menggunakan get(), bukan menggunakan index.

jika menggunakan index, ketika key tidak ada maka akan error, sedangkan jika dengan get() itu akan mengembalikan nilai default yaitu 0

```
1 class A():
2     attrClass = ''
3     def __init__(self):
4         self.name = 'empty'
5
6 print(A.__dict__)
7 print(len(A.__dict__))
8 print(A().__dict__)
9 print(len(A().__dict__))

{'__module__': '__main__', 'attrClass':
6
{'name': 'empty'}
1
```

A -> kelas  
A() -> instance

```
1 class Dog():
2     def __init__(self, name = 'none'):
3         self.name = name
4
5     def set(self, name = 'any'):
6         self.name = name
7
8 obj1 = Dog()
9 obj2 = obj1
10 obj2.set()
11
12 print(obj1.name)

any
```

obj2 merubah value obj1 karena merujuk pada memori yang sama

```
1 class A():
2     def __init__(self):
3         self.c = 'any'
4
5 hasattr(A(), 'c')
```

A() adalah instance jadi memiliki c

Cara baca:

Apakah A() memiliki attr 'c'

True

**Soal**

```
1 def a():
2     hey = 5
3
4     def b():
5         print(hey)
6
7     return b
8
9 x = a()
10 print(x())
```

5  
None

**Keterangan**

karena tidak ada return maka default mengembalikan None

```
1 def myFunc():
2     print('hello')
3
4 print(myFunc())
```

hello  
None

**hanya berlaku di instance (obj)**

Mengecek apakah suatu **objek** adalah instance dari suatu kelas atau subclass.

isinstance(obj, cls)

**hanya berlaku di kelas**

Mengecek apakah suatu kelas adalah subclass dari kelas lain.

What is the output of the following code?

```
class Ham :
    def __init__(self):
        self.v1 = 1
class Spam(Ham):
    def __init__(self):
        self.v2 = 2
s = Spam()
print(s.v1,s.v2)
()
```

0 2  
1 2  
Invalid Syntax  
AttributeError: 'Spam' object has no attribute 'v1'

A 

ketika mendeklarasikan `__init__` sendiri maka untuk memunculkan attr pada kelas induk harus menggunakan `super().__init__()` atau `Ham.__init__(self)`

## Soal



```
1 def foo(self, p):
2     print('Hello', p)
3
4 class Spam:
5     bar = foo
6
7 s = Spam()
8 s.bar('World!')
```

→ Hello World!

## Keterangan

foo disimpan ke bar tanpa di eksekusi, hanya menyimpan referensi

sedangkan jika foo() akan error karena ketika menyimpan ke bar, foo() akan ter eksekusi

```
1 class X : pass
2 class Y : pass
3 class Z (X, Y): pass
4
5 issubclass(Z, (list, X, Y))
```

True

Apakah Z adalah salah satu subclass dari tuple(list, X, Y)

- bukan subclass dari list
- subclass dari X
- subclass dari Y

can't duplicate

```
1 class A: pass
2 class B (A, A): pass

TypeError                                 Traceback (most recent call last)
<ipython-input-7-dda9e50ea591> in <cell line: 2>()
      1 class A: pass
----> 2 class B (A, A): pass

TypeError: duplicate base class A
```

setiap instance memiliki attr sendiri, ketika tidak mengoverwritte attr kelas maka akan selalu menggunakan value pada attr kelas

```
1 class MyClass:
2     foo = 100
3     def __init__(self):
4         self.bar = []
5
6     def add(self, p):
7         self.bar.append(p)
8
9 obj1, obj2 = MyClass(), MyClass()
10 obj1.add('Spam')
11 obj2.add('Ham')
12
13 obj2.foo = 200 #obj2 memiliki attr foo sendiri
14 #obj1 masih menggunakan attr foo MyClass
15
16 MyClass.foo = 300
17
18 print(obj1.bar, obj1.foo, obj2.bar, obj2.foo)
19

['Spam'] 300 ['Ham'] 200
```

## Miscellaneous

## Soal

```
1 import calendar
2
3 c = calendar.Calendar()
4
5 for weekday in c.iterweekdays():
6     print(weekday, end=" ")
7
8
0 1 2 3 4 5 6
```

## Keterangan

akan menghasilkan dari 0

```
1 from datetime import date, datetime
2
3 date_1 = date(1992, 1, 16)
4 date_2 = date(1991, 2, 5)
5
6 print(date_1 - date_2)
7
8 datetime1 = datetime(1992, 1, 16, 11, 11, 11)
9 datetime2 = datetime(1991, 2, 5)
10
11 print(datetime1 - datetime2)

345 days, 0:00:00
345 days, 11:11:11
```

jika hari = 0, maka tidak muncul

```
1 from datetime import date, datetime
2
3 date_1 = date(1992, 1, 16)
4 date_2 = date(1991, 2, 5)
5
6 print(date_1 - date_2)
7
8 datetime1 = datetime(1992, 1, 16, 11, 11, 11)
9 datetime2 = datetime(1992, 1, 16)
10
11 print(datetime1 - datetime2)

345 days, 0:00:00
11:11:11
```

bakal diurutkan sesuai abjad

```
['large', 'medium', 'small']
```

dengan urutan abjad :

```
a, b, ..., l, m, ..., s, ..., z
```

```
1 import os
2
3 os.mkdir('thumbnails')
4 os.chdir('thumbnails')
5
6 sizes = ['small', 'medium', 'large']
7
8 for size in sizes:
9     os.mkdir(size)
10
11 print(os.listdir())
12
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
(.venv) naufalnashif@MacBook-Air-Naufal python-
/Users/naufalnashif/Desktop/python-dev/main.py
['large', 'medium', 'small']
(.venv) naufalnashif@MacBook-Air-Naufal python-
```

```
1 import os
2
3 os.mkdir('pictures')
4 os.chdir('pictures')
5 os.mkdir('thumbnails')
6 os.chdir('thumbnails')
7 os.mkdir('tmp')
8 os.chdir('../')
9
10 print(os.getcwd())
```

```
/content/pictures
```

sudah kembali ke path /pictures dengan perintah  
os.chdir('../')

## Soal

Select the **true** statements. (Select **two** answers)

The `Lambda` function can accept a **maximum of two** arguments

The `Lambda` function can accept **any number** of arguments

The `Lambda` function can evaluate **multiple** expressions

The `Lambda` function can evaluate **only one** expression

```
1 with open('file.txt', 'w') as f:  
2     f.write("Hello\n")  
3     f.write("World!")  
4  
5 print('with for:')  
6 for x in open('file.txt', 'rt'):  
7     print(x)  
8 print()  
9 print('-----')  
10 print('with read:')  
11 with open('file.txt', 'r') as f:  
12     print(f.read(1))  
  
with for:  
Hello  
  
World!  
  
-----  
with read:  
H
```

## Keterangan

lambda arguments (bisa banyak): expression  
(only one)

`lambda x , y, z : x + y + z`

menggunakan `for` akan membaca line by line sama seperti `.readline`

sedangkan menggunakan `.read` akan membaca karakter demi karakter

- `.readline()` : Membaca line demi line hingga karakter newline atau EOF.
- `.read(1)` : Membaca karakter demi karakter.
- `.readlines()` : Membaca semua baris sekaligus, mengembalikannya sebagai list of strings.

What information can be read using the `uname` function provided by the `os` module? (Select **two** answers)

Last login date

Hardware identifier

Operating system name

Current path

`os.uname` hanya ada di unix/linux

Ya, data biner yang diberikan (**Hexadecimal**): `0x01 0x02 0x03 0x04` terdiri dari 4 byte.

• Penjelasan:

- Setiap nilai dalam format **hexadecimal** seperti `0x01`, `0x02`, `0x03`, dan `0x04` mewakili 1 byte.
- Total ada 4 nilai, sehingga ukurannya adalah **4 byte**.
- Dalam Python, Anda dapat membuat buffer yang sesuai dengan ukuran ini menggunakan `bytearray(4)`.

`bytearray(4)` -> membuat array dengan panjang 4 byte

Thank You !!