python面试题：

Python 新手在谋求一份 Python 编程工作前，必须熟知 Python 的基础知识。编程网站 DataFlair 的技术团队分享了一份 2018 年最常见 Python 面试题合集，既有基本的 Python 面试题，也有高阶版试题来指导你准备面试，试题均附有答案。面试题内容包括编码、数据结构、脚本撰写等话题。

Q 1：Python 有哪些特点和优点？

作为一门编程入门语言，Python 主要有以下特点和优点：

- 可解释
- 具有动态特性
- 面向对象
- 简明简单
- 开源
- 具有强大的社区支持

当然，实际上 Python 的优点远不止如此，可以阅读该文档，详细了解：

https://data-flair.training/blogs/python-tutorial/

Q 2：深拷贝和浅拷贝之间的区别是什么？

答：深拷贝就是将一个对象拷贝到另一个对象中，这意味着如果你对一个对象的拷贝做出改变时，不会影响原对象。在Python中，我们使用函数deepcopy()执行深拷贝，导入模块copy，如下所示：

```
1>>> import copy
2>>> b=copy.deepcopy(a)
```

而浅拷贝则是将一个对象的引用拷贝到另一个对象上，所以如果我们在拷贝中改动，会影响到原对象。我们使用函数function()执行浅拷贝，使用如下所示：

```
1>>> b=copy.copy(a)
```

Q 3. 列表和元组之间的区别是？

答：二者的主要区别是列表是可变的，而元组是不可变的。举个例子，如下所示：

```
1>>> mylist=[1,3,3]
2>>> mylist[1]=2
3>>> mytuple=(1,3,3)
4>>> mytuple[1]=2
5Traceback (most recent call last):
6File "<pyshell#97>", line 1, in <module>
7mytuple[1]=2
```

会出现以下报错：

```
TypeError: ‘tuple’ object does not support item assignment
```

关于列表和元组的更多内容，可以查看这里：

https://data-flair.training/blogs/python-tuples-vs-lists/

从Q4到Q20都是针对新手的Python面试基础试题，不过有经验的人也可以看看这些问题，复习一下基础概念。

Q 4. 解释一下 Python 中的三元运算子

不像 C++，我们在 Python 中没有?:，但我们有这个：

```
[on true] if [expression] else [on false]
```

如果表达式为True，就执行[on true]中的语句。否则，就执行[on false]中的语句。

下面是使用它的方法：

```
>>> a,b=2,3
>>> min=a if a<b else b
>>> min
```

运行结果：

```
2
>>> print("Hi") if a<b else print("Bye")
```

运行结果：

```
Hi
```

Q 5. 在 Python 中如何实现多线程？

一个线程就是一个轻量级进程，多线程能让我们一次执行多个线程。我们都知道，Python 是多线程语言，其内置有多线程工具包。

Python 中的 GIL（全局解释器锁）确保一次执行单个线程。一个线程保存 GIL 并在将其传递给下个线程之前执行一些操作，这会让我们产生并行运行的错觉。但实际上，只是线程在 CPU 上轮流运行。当然，所有的传递会增加程序执行的内存压力。

Q 6. 解释一下 Python 中的继承

当一个类继承自另一个类，它就被称为一个子类/派生类，继承自父类/基类/超类。它会继承/获取所有类成员（属性和方法）。

继承能让我们重新使用代码，也能更容易的创建和维护应用。Python 支持如下种类的继承：

- 单继承：一个类继承自单个基类
- 多继承：一个类继承自多个基类
- 多级继承：一个类继承自单个基类，后者则继承自另一个基类
- 分层继承：多个类继承自单个基类
- 混合继承：两种或多种类型继承的混合

更多关于继承的内容，参见：

https://data-flair.training/blogs/python-inheritance/

Q 7. 什么是 Flask?

Flask 是 Python 编写的一款轻量级 Web 应用框架。其 WSGI 工具箱采用 Werkzeug ，模板引擎则使用 Jinja2。Flask 使用 BSD 授权。其中两个环境依赖是 Werkzeug 和 jinja2，这意味着它不需要依赖外部库。正因如此，我们将其称为轻量级框架。

Flask 会话使用签名 cookie 让用户查看和修改会话内容。它会记录从一个请求到另一个请求的信息。不过，要想修改会话，用户必须有密钥 Flask.secret_key。

Q 8. 在 Python 中是如何管理内存的？

Python 有一个私有堆空间来保存所有的对象和数据结构。作为开发者，我们无法访问它，是解释器在管理它。但是有了核心 API 后，我们可以访问一些工具。Python 内存管理器控制内存分配。

另外，内置垃圾回收器会回收使用所有的未使用内存，所以使其适用于堆空间。

Q 9. 解释 Python 中的 help() 和 dir() 函数

Help() 函数是一个内置函数，用于查看函数或模块用途的详细说明：

```
1>>> import copy
2>>> help(copy.copy)
```

运行结果为：

```
1Help on function copy in module copy:
2
3
4copy(x)
5
6Shallow copy operation on arbitrary Python objects.
7
8See the module's __doc__ string for more info.
```

Dir() 函数也是 Python 内置函数，dir() 函数不带参数时，返回当前范围内的变量、方法和定义的类型列表；带参数时，返回参数的属性、方法列表。

以下实例展示了 dir 的使用方法：

```
1>>> dir(copy.copy)
```

运行结果为：

```
1['__annotations__', '__call__', '__class__', '__closure__', '__code__', '__defaults__', '__delattr__', '__dict__', '__dir__', '__doc__',
'__eq__', '__format__', '__ge__', '__get__', '__getattribute__', '__globals__', '__gt__', '__hash__', '__init__', '__init_subclass__',
'__kwdefaults__', '__le__', '__lt__', '__module__', '__name__', '__ne__', '__new__', '__qualname__', '__reduce__', '__reduce_ex__',
'__repr__', '__setattr__', '__sizeof__', '__str__', '__subclasshook__']
```

Q 10. 当退出 Python 时，是否释放全部内存？

答案是 No。循环引用其它对象或引用自全局命名空间的对象的模块，在 Python 退出时并非完全释放。

另外，也不会释放 C 库保留的内存部分。

Q 11. 什么是猴子补丁？

在运行期间动态修改一个类或模块。

```
1>>> class A:
2    def func(self):
3        print("Hi")
4>>> def monkey(self):
5print "Hi, monkey"
6>>> m.A.func = monkey
7>>> a = m.A()
8>>> a.func()
```

运行结果为：

```
1Hi, Monkey
```

Q 12. Python 中的字典是什么？

字典是 C++ 和 Java 等编程语言中所没有的东西，它具有键值对。

```
1>>> roots={25:5,16:4,9:3,4:2,1:1}
2>>> type(roots)
3<class 'dict'>
4>>> roots[9]
```

运行结果为：

```
13
```

字典是不可变的，我们也能用一个推导式来创建它。

```
1>>> roots={x**2:x for x in range(5,0,-1)}
2>>> roots
```

运行结果：

```
1{25: 5, 16: 4, 9: 3, 4: 2, 1: 1}
```

Q 13. 请解释使用 *args 和 **kwargs 的含义

当我们不知道向函数传递多少参数时，比如我们向传递一个列表或元组，我们就使用*args。

```
1>>> def func(*args):
2    for i in args:
3        print(i)
4>>> func(3,2,1,4,7)
```

运行结果为：

```
13
2
32
4
51
6
74
8
97
```

在我们不知道该传递多少关键字参数时，使用**kwargs来收集关键字参数。

```
1>>> def func(**kwargs):
2    for i in kwargs:
3        print(i,kwargs[i])
4>>> func(a=1,b=2,c=7)
```

运行结果为：

```
1a.1
2
3b.2
4
5c.7
```

Q 14. 请写一个Python逻辑，计算一个文件中的大写字母数量

```
1>>> import os
2
3>>> os.chdir('C:\Users\lifei\Desktop')
4>>> with open('Today.txt') as today:
5    count=0
6    for i in today.read():
7        if i.isupper():
8            count+=1
9print(count)
```

运行结果：

```
126
```

Q 15. 什么是负索引？

我们先创建这样一个列表：

```
1>>> mylist=[0,1,2,3,4,5,6,7,8]
```

负索引和正索引不同，它是从右边开始检索。

```
1>>> mylist[-3]
```

运行结果：

```
16
```

它也能用于列表中的切片：

```
1>>> mylist[-6:-1]
```

结果：

```
1[3, 4, 5, 6, 7]
```

Q 16. 如何以就地操作方式打乱一个列表的元素？

为了达到这个目的，我们从random模块中导入shuffle()函数。

```
1>>> from random import shuffle
2>>> shuffle(mylist)
3>>> mylist
```

运行结果：

```
1[3, 4, 8, 0, 5, 7, 6, 2, 1]
```

Q 17. 解释Python中的join()和split()函数

```
1Join()能让我们将指定字符添加至字符串中。
2
3>>> ','.join('12345')
```

运行结果：

```
1‘1,2,3,4,5’
```

Split()能让我们用指定字符分割字符串。

```
1>>> '1,2,3,4,5'.split(',')
```

运行结果：

```
1[‘1’, ‘2’, ‘3’, ‘4’, ‘5’]
```

Q 18. Python区分大小写吗？

如果能区分像myname和Myname这样的标识符，那么它就是区分大小写的。也就是说它很在乎大写和小写。我们可以用Python试一试：

```
1>>> myname='Ayushi'
2>>> Myname
3Traceback (most recent call last):
4File "<pyshell#3>", line 1, in <module>
```

运行结果：

```
1Myname
2NameError: name ‘Myname’ is not defined
```

可以看到，这里出现了NameError，所以Python是区分大小写的。

Q 19. Python中的标识符长度能有多长？

在Python中，标识符可以是任意长度。此外，我们在命名标识符时还必须遵守以下规则：

1. 只能以下划线或者 A-Z/a-z 中的字母开头
2. 其余部分可以使用 A-Z/a-z/0-9
3. 区分大小写
4. 关键字不能作为标识符，Python中共有如下关键字：

Q 20. 怎么移除一个字符串中的前导空格？

字符串中的前导空格就是出现在字符串中第一个非空格字符前的空格。我们使用方法lstrip()可以将它从字符串中移除。

```
1>>> '   Ayushi '.lstrip()
```

结果：

```
1‘Ayushi   ‘
```

可以看到，该字符串既有前导字符，也有后缀字符，调用lstrip()去除了前导空格。如果我们想去除后缀空格，就用rstrip()方法。

```
1>>> '   Ayushi '.rstrip()
```

结果：

```
1‘   Ayushi’
```

进阶版Python面试题。

Q 21. 怎样将字符串转换为小写？

我们使用lower()方法。

```
1>>> 'AyuShi'.lower()
```

结果：

```
1‘ayushi’
```

使用upper()方法可以将其转换为大写。

```
1>>> 'AyuShi'.upper()
```

结果：

```
1‘AYUSHI’
```

另外，使用isupper()和islower()方法检查字符串是否全为大写或小写。

```
1>>> 'AyuShi'.isupper()
2False
3
4>>> 'AYUSHI'.isupper()
5True
6
7>>> 'ayushi'.islower()
8True
```

```
 9
10>>> '@yu$hi'.islower()
11True
12
13>>> '@YU$HI'.isupper()
14True
```

那么，像@和$这样的字符既满足大写也满足小写。

Istitle()能告诉我们一个字符串是否为标题格式。

```
1>>> 'The Corpse Bride'.istitle()
2True
```

Q 22. Python中的pass语句是什么？

在用Python写代码时，有时可能还没想好函数怎么写，只写了函数声明，但为了保证语法正确，必须输入一些东西，在这种情况下，我们会使用pass语句。

```
1 >>> def func(*args):
2           pass
3>>>
```

同样，break语句能让我们跳出循环。

```
1>>> for i in range(7):
2    if i==3: break
3print(i)
```

结果：

```
10
2
31
4
52
```

最后，continue语句能让我们跳到下个循环。

```
1>>> for i in range(7):
2    if i==3: continue
3print(i)
```

结果：

```
 10
 2
 31
 4
 52
 6
 74
 8
 95
10
116
```

Q 23. Python中的闭包是什么？

当一个嵌套函数在其外部区域引用了一个值时，该嵌套函数就是一个闭包。其意义就是会记录这个值。

```
1>>> def A(x):
2    def B():
3        print(x)
4    return B
5>>> A(7)()
```

结果：

```
17
```

更多关于闭包的知识，请参看这里：

https://data-flair.training/blogs/python-closure/

Q 24. 解释一下Python中的//，%和 ** 运算符

```
1//运算符执行地板除法（向下取整除），它会返回整除结果的整数部分。
2
3>>> 7//2
43
```

这里整除后会返回3.5。

同样地，**执行取幂运算。a**b会返回a的b次方。

```
1>>> 2**10
21024
```

最后，%执行取模运算，返回除法的余数。

```
1>>> 13%7
26
3>>> 3.5%1.5
40.5
```

Q 24. 在Python中有多少种运算符？解释一下算数运算符。

在Python中，我们有7种运算符：算术运算符、关系运算符、赋值运算符、逻辑运算符、位运算符、成员运算符、身份运算符。

我们有7个算术运算符，能让我们对数值进行算术运算：

1.加号（+），将两个值相加

```
1>>> 7+8
215
```

2.减号（-），将第一个值减去第二个值

```
1>>> 7-8
2-1
```

3.乘号（*），将两个值相乘

```
1>>> 7*8
256
```

4.除号（/），用第二个值除以第一个值

```
1>>> 7/8
20.875
3>>> 1/1
41.0
```

5.向下取整除、取模和取幂运算，参见上个问题。

Q 25. 解释一下Python中的关系运算符

关系运算符用于比较两个值。 1.小于号（<），如果左边的值较小，则返回True。

```
1>>> 'hi'<'Hi'
2False
```

2.大于号（>），如果左边的值较大，则返回True。

```
1>>> 1.1+2.2>3.3
2True

3.小于等于号（<=），如果左边的值小于或等于右边的值，则返回Ture。
```

1>>> 3.0<=3 2True

```
4.大于等于号（>=），如果左边的值大于或等于右边的值，则返回True。
```

1>>> True>=False 2True

```
1. 等于号（==），如果符号两边的值相等，则返回True。
```

1>>> {1,3,2,2}=={1,2,3} 2True

```
1. 不等于号（!=），如果符号两边的值不相等，则返回True。
```

1>>> True!=0.1 2True 3>>> False!=0.1 4True

```
### **Q 26. 解释一下Python中的赋值运算符**
这在Python面试中是个重要的面试问题。

我们将所有的算术运算符和赋值符号放在一起展示：
```

1>>> a=7 2>>> a+=1 3>>> a 48 5 6>>> a-=1 7>>> a 87 9 10>>> a*=2 11>>> a 1214 13 14>>> a/=2 15>>> a 167.0 17 18>>> a**=2 19>>> a 2049 21 22>>> a//=3 23>>> a 2416.0 25 26>>> a%=4 27>>> a 280.0

```
### **Q 27. 解释一下Python中的逻辑运算符**

Python中有3个逻辑运算符：and，or，not。
```

1>>> False and True 2False 3 4>>> 7<7 or True 5True 6 7>>> not 2==2 8False

Q 28. 解释一下Python中的成员运算符

通过成员运算符‘in’和‘not in’，我们可以确认一个值是否是另一个值的成员。

1>>> 'me' in 'disappointment' 2True 3 4>>> 'us' not in 'disappointment' 5True

Q 29. 解释一下Python中的身份运算符

这也是一个在Python面试中常问的问题。

通过身份运算符‘is’和‘is not’，我们可以确认两个值是否相同。

1>>> 10 is '10' 2False 3 4>>> True is not False 5True

Q 30. 讲讲Python中的位运算符

该运算符按二进制位对值进行操作。

1. 与（&），按位与运算符：参与运算的两个值,如果两个相应位都为1,则该位的结果为1,否则为0

```python
1>>> 0b110 & 0b010
22
```

2.或（|），按位或运算符：只要对应的二个二进位有一个为1时，结果位就为1。

```
1>>> 3|2
23
3
```

3.异或（^），按位异或运算符：当两对应的二进位相异时，结果为1

```
1>>> 3^2
21
```

4.取反（~），按位取反运算符：对数据的每个二进制位取反,即把1变为0,把0变为1

```
1>>> ~2
2-3
```

5.左位移（<<），运算数的各二进位全部左移若干位，由 << 右边的数字指定了移动的位数，高位丢弃，低位补0

```
1>>> 1<<2
24
```

6.右位移（>>），把">>"左边的运算数的各二进位全部右移若干位，>> 右边的数字指定了移动的位数

```
1>>> 4>>2
21
```

更多关于运算符的知识，参考这里： ***https://data-flair.training/blogs/python-operators/***

Q 31. 在Python中如何使用多进制数字？

我们在Python中，除十进制外还可以使用二进制、八进制和十六进制。

1. 二进制数字由0和1组成，我们使用 0b 或 0B 前缀表示二进制数。

```
1>>> int(0b1010)
```

```
210
```

2.使用bin()函数将一个数字转换为它的二进制形式。

```
1>>> bin(0xf)
2‘0b1111’
```

3.八进制数由数字 0-7 组成，用前缀 0o 或 0O 表示 8 进制数。

```
1>>> oct(8)
2‘0o10’
```

4.十六进数由数字 0-15 组成，用前缀 0x 或者 0X 表示 16 进制数。

```
1>>> hex(16)
2‘0x10’
3
4>>> hex(15)
5‘0xf’
```

Q 32. 怎样获取字典中所有键的列表？

使用 keys() 获取字典中的所有键

```
1>>> mydict={'a':1,'b':2,'c':3,'e':5}
2>>> mydict.keys()
3dict_keys(['a', 'b', 'c', 'e'])
```

Q 33. 为何不建议以下划线作为标识符的开头

因为Python并没有私有变量的概念，所以约定速成以下划线为开头来声明一个变量为私有。所以如果你不想让变量私有，就不要使用下划线开头。

Q 34. 怎样声明多个变量并赋值？

一共有两种方式：

```
1>>> a,b,c=3,4,5 #This assigns 3, 4, and 5 to a, b, and c respectively
2>>> a=b=c=3 #This assigns 3 to a, b, and c
```

Q 35. 元组的解封装是什么？

首先我们来看解封装：

```
1>>> mytuple=3,4,5
2>>> mytuple
3(3, 4, 5)
```

这将 3，4，5 封装到元组 mytuple 中。

现在我们将这些值解封装到变量 x，y，z 中：

```
1>>> x,y,z=mytuple
2>>> x+y+z
```

得到结果12.