第一天 环境部署和基本语法

一、 需求

假设目前需要写一个小的程序，程序的功能非常简单，就叫猜数游戏吧。

1. 给用户一个提示信息，让其输入一个数字
2. 接着拿用户输入的数字和 18 进行比较大小
3. 等于 18 , 就输出 "相等"
4. 小于 18，就输出 "小了"
5. 大于 18，就输出 "大了"

二、 需求分析和分解技术点

1. 程序和用户交互

思考一下，如何实现?

我们可以分析一下

给提示信息，让其输入一个数字

这里会用的和用户的交互，就是程序和用户的交互。

python 中 使用 `input` 函数实现

```
input("这里写提示信息，  必须使用引号引起来")
```

2. 变量

那用户的输入可以使用一个变量接收

```
n = input("请输入一个数字")
```

2.1 变量命名潜规则:

- 不要以单下划线和双下划线开头；如：_user或 __user
- 变量命名要易读；如：user_name,而不是username
- 不用使用标准库中(内置)的模块名或者第三方的模块名
- 不要用这些 Python 内置的关键字：

```
>>> import keyword
>>> keyword.kwlist
['False', 'None', 'True', 'and', 'as', 'assert',
'break', 'class', 'continue', 'def', 'del', 'elif',
'else', 'except', 'finally', 'for', 'from', 'global',
'if', 'import', 'in', 'is', 'lambda', 'nonlocal',
'not', 'or', 'pass', 'raise', 'return', 'try',
'while', 'with', 'yield']
```

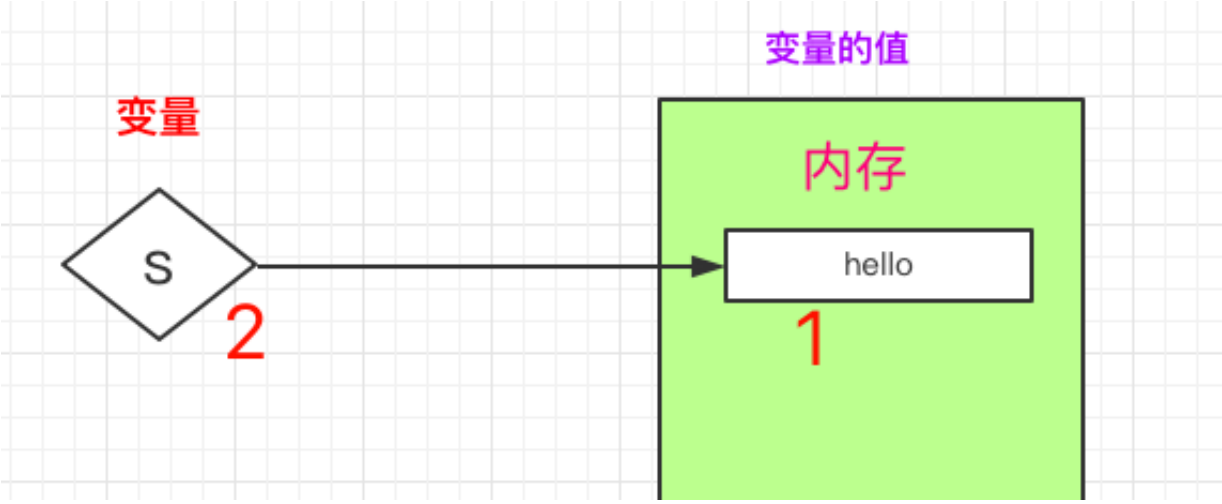

2.2 如何理解 python 的变量赋值

在 python 中究竟该如何正确理解变量的赋值过程呢?

```
s = 'hello'
```

以上的变量赋值，应该说成**把变量名 `s` 分配给 `hello` 这个对象更合理。**

`hello` 这个对象会在内存中先被创建，之后再把变量名 `s` 分配给这个对象。



所以要理解 Python 中的变量赋值，应该始终先看*等号右边*。

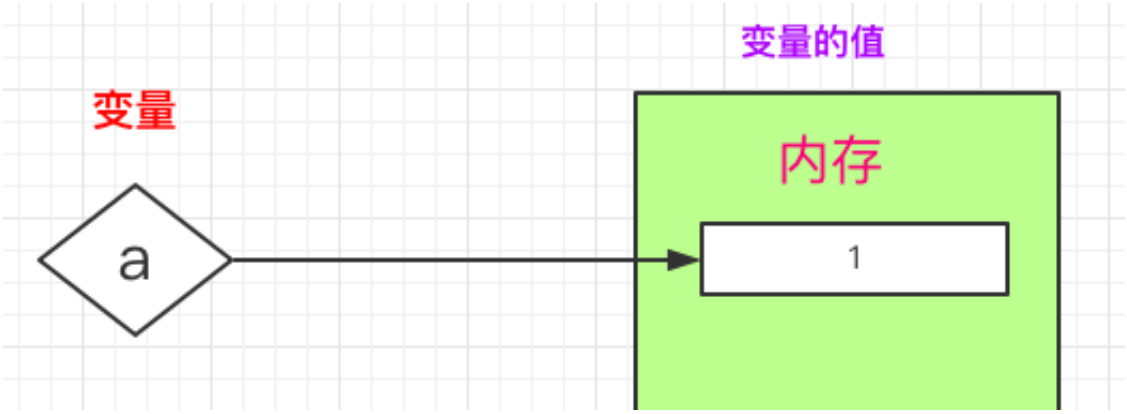
对象是在右边先被创建或者被获取，在此之后左边的变量名才会被绑定到对象上，这就像是为对象贴上了一个标签。

一个对象可以有多个标签或者名字。比如：我们自己就有很多名字，身份证上的名字，网络昵称等。

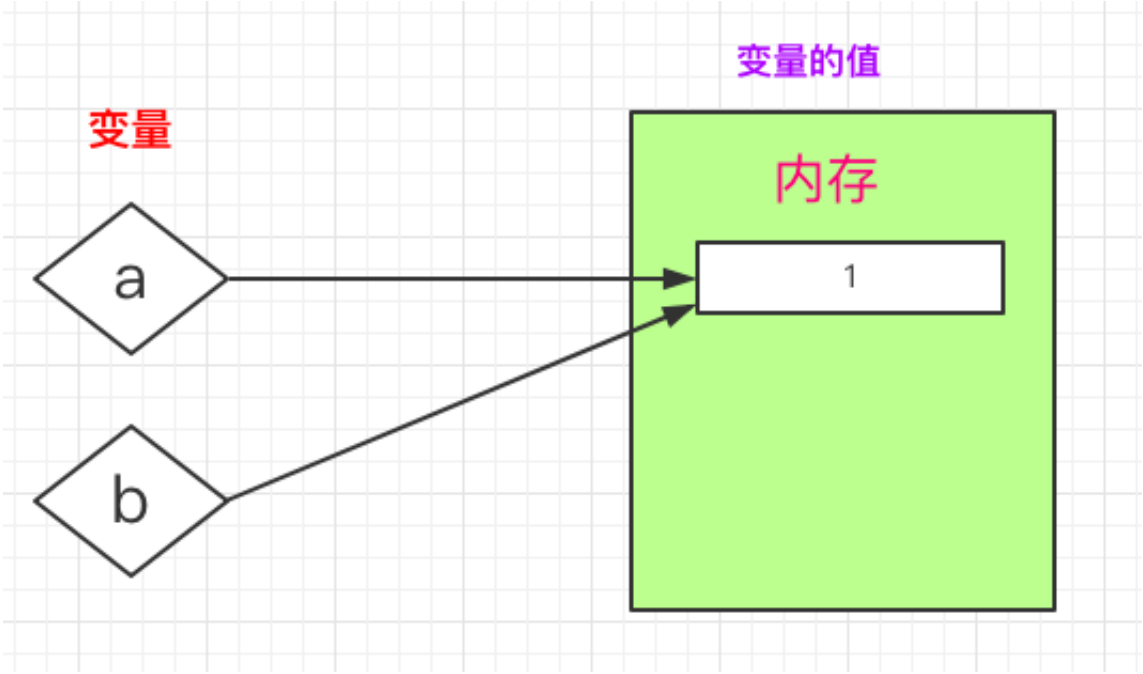
请看下面的代码示例：

```
a = 1
b = a
a = 2
print(b)  # b 会是 ?
```

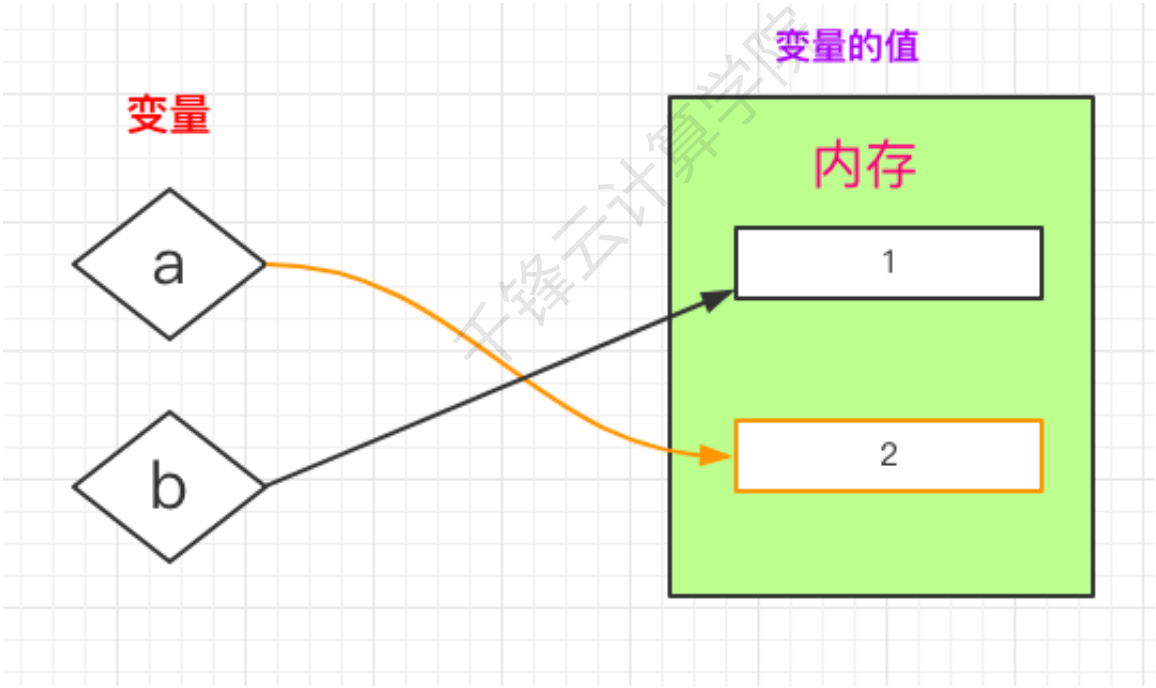
`a = 1` 时如下图：



`b = a` 时如下图:



`a = 2` 时如下图:



上面的 `b = a` 我们称它为 **传递引用**，此时对象会拥有两个名称(标签)，分别是 `a` 和 `b`

2.3 多元赋值

> 字符串以及后面讲的列表、元组都支持这种操作，也要元组解包

```
In [9]: n1, n2 = 1, 2

In [10]: n1
Out[10]: 1

In [11]: n2
Out[11]: 2
In [75]: s1, s2 = '12'

In [76]: s1
Out[76]: '1'

In [77]: s2
Out[77]: '2'

In [78]: num, s = [10, 'hello']   # 这个是列表，  后面会讲到

In [79]: num
Out[79]: 10

In [80]: s
Out[80]: 'hello'
```


好，至此，我们解决了使用变量来接收用户的输入，接下来就要解决判断的问题了。 那 python 中如何进行判断呢?

要判断需要使用判断条件

3. python 中的判断条件

```
>>> n = 10
>>> n == 10 # 等于
True                # 条件为真，则返回 True
```

```
>>> n != 10 # 不等于
False               # 条件为假，则返回 False
>>> n > 10  # 大于
False
>>> n < 10  # 小于
False
>>> n >= 10  # 大于等于
True
>>> n <= 10 # 小于等于
True
>>>
```

那接下来，在 Ipython 中来实际操作一下

```
In [1]: n = input("请输入一个数字>>:")
请输入一个数字>>:10

In [2]: n == 10
Out[2]: False

In [3]:
```


> 会发现返回 False 在编程语言中，数据是有类型之分的。`input()` 接收到的任何数据都会成为 字符串类型(str)，就是普通的字符串 而 我们等号 右边的 `10` 是整型(int)

4. 数据类型

4.1 查看数据的类型，使用 `type`

```
In [3]: type(n)    # 查看 input 接收到的数据类型
Out[3]: str

In [4]: type(10)
Out[4]: int

In [5]:
```

4.2 基本的数据类型

- 整型(int)

```
In [11]: type(0)
Out[11]: int

In [12]: type(-1)
Out[12]: int

In [13]: type(1)
Out[13]: int
```


- 浮点型（带小数点的小数）

```
In [17]: type(1.1)
Out[17]: float

In [18]: type(-0.1)
Out[18]: float

```

- 布尔型

```
In [19]: type(True)
Out[19]: bool

In [20]: type(False)
Out[20]: bool

```

- 字符串(str)

```
In [14]: type('10')
Out[14]: str

In [15]: type('hello')
Out[15]: str

In [16]: type('-1.1')
Out[16]: str

In [17]:

```


- 二进制(bytes)

```
In [18]: type(b'hello')
Out[18]: bytes
```

我们来验证一下 input 接收的数据的类型

```
In [36]: n = input("请输入一个数字>>:")
请输入一个数字>>:10

In [37]: n
Out[37]: '10'

In [38]: type(n)
Out[38]: str
```

要想把用户的输入(str)和整型(int)进行正确的比较大小，就需要把字符串类型的数据转换整型。 这种把一个数据从一个类型转换为另外一个类型的操作叫类型装换

5. 类型转换

- 转换为 int

```
In [21]: int('10')
Out[21]: 10

In [22]: int('-10')
Out[22]: -10

In [23]: int(1.9)
Out[23]: 1
```

- 转换为 float

```
In [25]: float(1)
Out[25]: 1.0

In [26]: float(-1)
Out[26]: -1.0

In [27]: float('1.1')
Out[27]: 1.1
```

- 转换为 str

```
In [28]: str(111)
Out[28]: '111'

In [29]: str(-111)
Out[29]: '-111'

In [30]: str(-11.1)
Out[30]: '-11.1'

In [31]: str(b'hello', encoding='utf-8')
Out[31]: 'hello'
```


> 二进制转换字符串的时候，需要指定字符编码

- 转换为二进制

```
In [32]: bytes('千锋', encoding=('utf-8'))
Out[32]: b'\xe5\x8d\x83\xe9\x94\x8b'

In [58]: b = bytes('千锋', encoding=('utf-8'))

In [59]: b
Out[59]: b'\xe5\x8d\x83\xe9\x94\x8b'

```

```
In [60]: str(b, encoding='utf-8')
Out[60]: '千锋'

In [61]: s= str(b, encoding='utf-8')

In [62]: s
Out[62]: '千锋'

In [63]: type(s)
Out[63]: str

```

> 注意字符串转二进制时候，需要指定字符编码

6. if 判断语句

判断条件可以用在 if 判断语句中 语法结构是这样的：

- 语法一：

```
if   判断条件：    # 冒号必须的
     如果判断条件为真，执行这里的代码，这里的代码必须缩进4个空格
     并且每一行代码的缩进要一致
```

示例：

```
In [39]: n = input("请输入一个数字>>:")
请输入一个数字>>:18

In [40]: n = int(n)

In [41]: if n == 18:
    ...:     print("相等")
    ...:
相等

```

- 语法二：

```
if  判断条件：
    如果判断条件为真，执行这里的代码
else:   # 这里的冒号也是必须的
    如果判断条件为假，执行这里的代码，这里的代码必须缩进4个空格
    并且每一行代码的缩进都要一致
```

示例：

```
In [44]: if n == 10:
    ...:     print("相等")
    ...: else:
    ...:     print("不相等")
    ...:
不相等

```

- 语法三：

```
if  判断条件：
    如果判断条件添加为真，执行这里的代码，这里的代码必须缩进4个空格
    并且每一行代码的缩进要一致
elif  判断条件：   # 这里同样需要加条件
    如果判断条件添加为真，执行这里的代码，这里的代码必须缩进4个空格
    并且每一行代码的缩进要一致
else:   # 这里的冒号也是必须的
    如果判断条件为假，执行这里的代码，这里的代码必须缩进4个空格
    并且每一行代码的缩进都要一致
```

> `elif` 根据需求可以出现多个

示例:

```
In [51]: n = 20

In [52]: if n == 10:
    ...:     print("相等")
    ...: elif n > 10:
    ...:     print("大了")
    ...: else:
    ...:     print("小了")
    ...:
大了

```


做实验，特结论

下面的代码会打印出几次 `'ok'`

```
if 1 == 1:
    print("ok")
elif  2 == 2:
    print("ok")
elif 3 == 3:
    print("ok")
```

三、Python 程序

在生产中，通常我们会把程序的代码写的一个文件种，这个文件就成为 Python 的一个程序文件，文件名一般都是以 `.py` 为结尾，有时候也成为 一个 python 的程序。

使用 `vi` 编辑器,来把我们这个猜数游戏的小程序写一下吧

```
#!/usr/bin/env python3
# file name hello.py

print("猜数游戏开始")

n = input("请输入一个数字")
n = int(n)

if n == 18:
    print("猜对了")
elif  n > 18:
    print("大了")
else:
    print("小了")
```


> 第一行不是注释，和 shell 脚本一样，是在声明这个脚本默认使用的解释器

执行 python 程序

```
[root@qfedu.com ~]# ./hello.py
猜数游戏开始
请输入一个数字8
小了
```

四、 while 循环

语法：

```
while 条件表达式：
    条件表达示为真，就执行这里的代码，必须缩进 4 个空格
    多行代码保持缩进一致
```

> 条件表达式可以是:
>
> - `True` # 布尔值的 True
> - `1 < 10` # 凡是在 if 语句中使用的判断表达示，这里都可以使用

猜数游戏优化版本


```
#!/usr/bin/env python3

print("猜数游戏开始")

while True:
    n = input("请输入一个数字")

    # 如果输入空，就重新开始新一轮的循环
    if not n:
        continue

    # 如果输入 q 就是跳出循环
    if n == 'q':
```

```
        break

    n = int(n)

    if n == 18:
        print("猜对了")
    elif  n > 18:
        print("大了")
    else:
        print("小了")

# 退出循环后，程序继续运行下面的代码
exit("退出程序..")
```

五、函数的定义和调用

1. 函数的定义

```
def 函数名():
    """函数的说明，主要是说明一下函数的主要功能，这是可选的"""
    函数体，就是代码
    缩进 4 个空格，多行缩进保持一致
```

> 函数名的规则和变量名的命名规则一致

2. 函数的调用

调用方式:

```
函数名()
```

> python 属于解释性语言，就是代码需要读一行，解释器解释一行。 因此，函数就像是 定义一个变量，必须先定义函数，才能调用函数。

3. 示例

```
def  foo():
    print("我是函数体，只有在调用函数时，这里的代码才会被执行")

foo()
执行后会输出：
我是函数体，只有在调用函数时，这里的代码才会被执行
```

那我们现在可以把之前写的猜数游戏，编写函数

```
#!/usr/bin/env python3

def guess_number():
    """输入一个数字，和18 比较大小"""
    print("猜数游戏开始")

    while True:
        n = input("请输入一个数字")

        # 如果输入空，就重新开始新一轮的循环
        if not n:
            continue

        # 如果输入 q 就是跳出循环
        if n == 'q':
            break

        n = int(n)
```

```
19
20          if n == 18:
21              print("猜对了")
22          elif  n > 18:
23              print("大了")
24          else:
25              print("小了")
26
27
28  # 调用函数
29  guess_number()
30
31  exit("退出程序..")
```

六、今日作业：

编写一个小程序，实现如下效果

用户输入一个数字，返回对应的服务名称，加上没有对应的服务，就返回未知的服务。 输入 q 退出。


```
1   [root@qfedu.com ~]# python3 search_server.py
2   常用端口-->查询程序
3   请输入一个常用的服务默认端口号:80
4   HTTP 服务
5   请输入一个常用的服务默认端口号:22
6   SSHD 服务
7   请输入一个常用的服务默认端口号:21
8   FTP 服务
9   请输入一个常用的服务默认端口号:3306
10  Mysql 服务
11  请输入一个常用的服务默认端口号:9080
12  未知服务
13  请输入一个常用的服务默认端口号:
```