# 流程控制

1. **填空题**
2. \_\_\_if\_\_\_\_\_语句是最简单的条件语句。
3. Python中的循环语句有\_\_\_\_for\_\_\_\_和\_\_\_while\_\_\_\_\_循环。
4. 若循环条件的值变为\_\_\_True\_\_\_\_\_，说明程序进入无限循环。
5. \_\_\_\_for\_\_\_\_循环一般用于实现遍历循环。
6. \_\_\_\_\_continue\_\_\_语句可以跳出本次循环，执行下一次循环。
7. **判断题**
8. if-else语句可以处理多个分支条件。（×）
9. if语句不支持嵌套使用。（×）
10. elif可以单独使用。（ ×）
11. break语句用于结束循环。（×）
12. for循环只能遍历字符串。（×）
13. **选择题**
14. 下列选项中，运行后会输出1、2、3的是（C）

for i in range(3):

print(i)

for i in range(2):

print(i + 1)

nums = [0, 1, 2]

for i in nums:

print(i + 1)

i = 1

while i < 3:

print(i)

i = i + 1

1. 现有如下代码：

sum = 0

for i in range(100):

if(i % 10):

continue

sum = sum + i

print(sum)

若运行代码，输出的结果为（C）。

1. 5050
2. 4950
3. 450
4. 45
5. 已知x=10，y=20，z=30；以下代码执行后x、y、z的值分别为（C ）。

if x < y:

z = x

x = y

y = z

1. 10，20，30
2. 10，20，20
3. 20，10，10
4. 20，10，30
5. 已知x与y的关系如表3-1所示：

表3-1 x与y的关系

|  |  |
| --- | --- |
| **x** | **y** |
| x<0 | x−1 |
| x=0 | x |
| x>0 | x+1 |

以下选项中，可以正确地表达x与y之间关系的是（C ）。

y = x + 1

if x >= 0:

if x == 0:

y = x

else:

y = x - 1

y = x - 1

if x! = 0:

if x > 0:

y = x + 1

else:

y = x

if x <= 0:

if x < 0:

y = x - 1

else:

y = x

else:

y = x + 1

y = x

if x <= 0:

if x < 0:

y = x - 1

else:

y = x + 1

1. 下列语句中，可以跳出循环结构的是（A ）。
2. continue
3. break
4. if
5. while
6. **简答题**
7. 简述break和continue的区别。

break语句用于终止当前循环或者switch语句，并跳出循环或者switch语句的执行。当程序执行到break语句时，程序会立即跳出当前循环或者switch语句，继续执行循环或者switch语句之后的代码。break语句通常用于在满足某个条件时提前结束循环。

continue语句用于跳过当前循环中剩余的代码，并继续执行下一次循环。当程序执行到continue语句时，程序会立即跳过continue语句之后的代码，直接进入下一次循环的判断条件。continue语句通常用于在某些条件下跳过当前循环的执行，但不终止整个循环。

1. 简述while和for语句的区别。

while语句会在每次循环开始之前先判断条件是否满足，只有当条件为真时，才会执行循环体中的代码。如果条件为假，则跳过循环体，直接执行循环之后的代码。while语句适用于不确定循环次数的情况，可以根据条件动态控制循环的执行。

for语句由三个部分组成：初始化、条件和更新。在循环开始之前，先执行初始化部分的代码；然后在每次循环开始之前判断条件是否满足，只有当条件为真时，才会执行循环体中的代码；最后在每次循环结束之后执行更新部分的代码。for语句适用于已知循环次数的情况，可以在一个语句中完成循环的初始化、条件判断和更新操作。