

## 实例 9：九九乘法表

乘法口诀是中国古代筹算中进行乘法、除法、开方等运算的基本计算规则，沿用至今已有一千多年。古代的乘法口诀与现在使用的乘法口诀顺序相反，自上而下从“九九八十一”开始到“一一如一”为止，因此，古人用乘法口诀的前两个字“九九”做为此口诀的名称。

本实例要求编写程序，实现通过 for 循环嵌套输出下列样式的九九乘法表的功能。

```
1*1=1
1*2=2   2*2=4
1*3=3   2*3=6   3*3=9
1*4=4   2*4=8   3*4=12  4*4=16
1*5=5   2*5=10  3*5=15  4*5=20  5*5=25
1*6=6   2*6=12  3*6=18  4*6=24  5*6=30  6*6=36
1*7=7   2*7=14  3*7=21  4*7=28  5*7=35  6*7=42  7*7=49
1*8=8   2*8=16  3*8=24  4*8=32  5*8=40  6*8=48  7*8=56  8*8=64
1*9=9   2*9=18  3*9=27  4*9=36  5*9=45  6*9=54  7*9=63  8*9=72  9*9=81
```

### 实例目标

- 掌握 while 循环的使用
- 了解 break 的基本用法

### 实例分析

九九乘法表一共有九行，每行等式的变量和行号相等，例如第二行包含两个等式，第六行包含 6 个等式，以此类推，第九行包含 9 个等式。根据其特点可知可使用 for 循环嵌套解决此问题。

我们可以定义变量 i 控制乘法表的行数与变量 j 控制乘法表等式量的输出。

第一个 for 循环用来控制乘法表中每行的第一个因子和表的行数；第二个 for 循环中变量 j 取值范围的确定建立在第一个 for 循环的基础上，它的取值是第一个 for 循环中变量的值，换言之，j 的取值根据行数变化，运行到第几行，j 的最大值就是几。

为了控制格式，将乘法表分行，需要在每行的末尾输出一个换行。

### 代码实现

```
for i in range(1, 10):
    for j in range(1, i + 1):
        print(str(j) + str("*") + str(i) + "=" + str(i * j), end="\t")
```

```
print() # 换行输出
```

上述代码中，第 1 个 for 循环的循环因子 i 的通过 range() 函数设置，其取值范围为 1-9。因为等式的数量与行号相等，所以在第 2 个 for 循环中变量 j 最大取值范围为等式数量。行数与等式量控制好后，便可以对乘法表中的乘法口诀进行拼接，拼接完成后进行换行输出。

## 代码测试

运行代码，控制台输出结果如下：

```
1*1=1
1*2=2  2*2=4
1*3=3  2*3=6  3*3=9
1*4=4  2*4=8  3*4=12  4*4=16
1*5=5  2*5=10  3*5=15  4*5=20  5*5=25
1*6=6  2*6=12  3*6=18  4*6=24  5*6=30  6*6=36
1*7=7  2*7=14  3*7=21  4*7=28  5*7=35  6*7=42  7*7=49
1*8=8  2*8=16  3*8=24  4*8=32  5*8=40  6*8=48  7*8=56  8*8=64
1*9=9  2*9=18  3*9=27  4*9=36  5*9=45  6*9=54  7*9=63  8*9=72  9*9=81
```