选14、数据结构与算法效率

1、算法效率

同一个问题可能会有不同的解决方法，也就是说可能有不同的算法。通常，算法效率的高低可由算法复杂度来衡量。

算法复杂度分为时间复杂度和空间复杂度。其中时间复杂度反映了算法执行所需的时间，而空间复杂度反映了算法执行所需占用的存储空间。

2、时间复杂度

时间复杂度常用符号O表述，称为大O记法。

对于问题：求1+2+···+n的和的问题，可以由两种算法实现：

算法1：

```
n=int(input())
s=(1+n)*n/2
print(s)
```

算法2：

```
n=int(input())
s=0
for i in range(1,n+1)
    s=s+i
print(s)
```

算法1执行了3条语句

算法2执行了1+1+n+n+1=3+2n条语句

随着n的增长，算法1还是执行3条语句，而算法2执行的语句会越来越多。当n趋向无穷大时，算法1执行的次数与1在同一数量级，而算法2的执行次数与n同一数量级。

因此，算法1的时间复杂度为O(1)，而算法2的时间复杂度为O(n)

再看以下代码：

```
n=int(input())
for i in range(n):
    for j in range(n):
        if i+j==(i*j)%(2*i+j):
            print(i,j)
```

以上程序共需执行的语句次数为：

```
n=int(input())----1次
for i in range(n):-----n次
    for j in range(n):------n*n次
        if i+j==(i*j)%(2*i+j):------n*n次
print(i,j)-----1次
```

共执行2+n+2*n^2次

当n趋向无穷时，执行次数与n^2在同一数量级。因此该算法的时间复杂度为O(n^2)

3、推导大O阶的方法

1）用常数1取代运行时间中所有加法的常数

2）只保留最高阶项

3）如果最高阶项存在且不是1，那么去除最高阶项的系数

如：

5+2n+3n^2------>2n+3n^2----->3n^2---->n^2

常见的时间复杂度耗时比较：

O(1)<O(log2(n))<O(n)<O(n^2)<O(n^3)<O(2^n)<O(n!)

tips：

请用matplotlib对这些常用的时间复杂度建立可视化图表

4、数据结构对算法效率的影响

数据组织成不同形式，是为了满足不同问题的需求，便于算法对数据的操作。

数组或者链表都能存储数据。如果从数据访问角度，数组可以直接根据下标来访问数据元素，对于长度为n的数组，其访问时间复杂度为O(1)，而链表需要从头结点访问，访问任意节点的平均所需次数为n/2次，因此其时间复杂度为O(n)；但如果从数据插入操作的角度看，对于数组，插入任意数据都要引起大量的数据移动，平均移动次数为n/2，因此时间复杂度为O(n)，而对于链表而言，只需完成两步操作，因此时间复杂度为O(1)

因此，在开发具体软件时，需要考虑哪种操作更为频繁，而选择哪种数据结构。