时空复杂度

By 吴羽晗.

一、语句

语句 是构成程序的基本单位。C++ 中的 **语句** 之间以分号、括号、关键字等隔开。

示例

```
int main() {

    for (int i = 1; i <= 3; ++ i) {
        printf("🍎");
    }

    return 0;
}
```

上述 `main()` 函数由 `int i = 1`，`i <= 3`，`++ i`，`printf("🍎")`，`return 0` 五个语句构成。

二、语句执行次数

C++ 程序运行时，每条 **语句** 的耗时基本相同。故 **语句执行次数** 可以大致反映程序的运行时间。

语句执行次数 记作 T。若 T 与变量 $x, y, \cdots$ 相关，则应记作函数形式 $T(x, y, \cdots)$。

示例 1

```
int func1() {

    printf("🍎");      // 执行 1 次
```

```
    return 0;          // 执行 1 次
}
```

调用一次 `func1()`，共执行 2 次语句，即 $T = 2$。

示例 2

```
void func2(int n) {

    for (
        int i = 1;  // 执行 1 次
        i <= n;     // 执行 n+1 次
        ++ i        // 执行 n 次
    ) {
        printf("🍎");    // 执行 n 次
    }
}
```

调用一次 `func2(n)`，共执行 $3n + 2$ 次语句，即 $T(n) = 3n + 2$。

三、时间复杂度

时间复杂度 是衡量程序效率的量度，它是 **语句执行次数** T 遵循以下规则的化简结果：

1. 常数项化为 1；
2. 各项的系数化为 1；
3. 只保留最高阶的项；
4. 忽略对数的底数，$\log_a n$ 应化为 $\log n$。

时间复杂度 记作 $O(f)$，其中 f 是 T 的化简结果。

> 化简的原则：关注主导因素，忽略次要细节。

示例 1

语句执行次数	时间复杂度
$T = 114514$	$O(1)$

语句执行次数	时间复杂度
$T(n) = 4n^2 + 3n + 1$	$O(n^2)$
$T(n) = n\sqrt{3n} + 3n$	$O(n\sqrt{n})$
$T(n) = 5n \log_3 n$	$O(n \log n)$

示例 2

```
void func3(int n) {

    for (int i = 1; i <= n; ++ i) {
        for (int j = 1; j <= n; ++ j) {
            printf("🔥");
        }
    }
    // 大约执行 n^2 次语句

    for (int i = 1; i <= n; ++ i) {
        printf("🔥");
    }
    // 大约执行 n 次语句
}
```

函数 `func3(n)` 的时间复杂度是 $O(n^2)$。

示例 3

```
void func4(int n) {

    for (int i = 1; i <= n; ++ i) {
        for (int j = i; j <= n; ++ j) {
            printf("🔥");
        }
    }
}
```

调用一次 `func4(n)`，语句执行次数约为

$$T(n) \approx n + (n-1) + (n-2) + \cdots + 1 = \frac{1}{2}n^2 + \frac{1}{2}n$$

时间复杂度是 $O(n^2)$。

示例 4

```
int func5(int n) {

    int i = 0, sum = 0;

    while (sum <= n) {
        i ++;
        sum += i;
    }

    return sum;
}
```

设 `while` 循环执行了 t 次，则

$$\begin{cases} \text{sum} = 1 + 2 + \cdots + t = \dfrac{t(t+1)}{2} \\ \text{sum} \leq n \end{cases}$$

$$\therefore t^2 + t \leq 2n$$

$$\therefore t \lessapprox \sqrt{2n}$$

即 `while` 循环大约执行了 $\sqrt{2n}$ 次。

函数 `func5(n)` 的时间复杂度是 $O(\sqrt{n})$。

时间复杂度的分类

分类	记号
最大（最坏）时间复杂度	$O(f)$
最小（最好）时间复杂度	$\Omega(f)$
精确时间复杂度	$\Theta(f)$

时间复杂度 默认指 **最大时间复杂度**。

四、空间复杂度

空间复杂度 是衡量程序内存占用的量度，它是 **变量定义次数** 的化简结果，采用和 **时间复杂度** 完全相同的记号和化简规则。

示例

```
void func6(int n) {
    int a;              // 定义 1 次
    char b[n];          // 数组含 n 个变量，定义 n 次
    float c[n][n];      // 数组含 n^2 个变量，定义 n^2 次
}
```

调用一次 `func6(n)`，共定义 $n^2 + n + 1$ 次变量，空间复杂度为 $O(n^2)$。