保定市信息学竞赛公益大讲堂 第五节

循环语句和认识算法

5.1 循环语句

循环语句就是让程序重复执行某些操作，直到满足某个条件。例如，我们可以用循环来让程序多次输出一行文字，或者计算多个数字的和。

5.1.1 while 循环

`while`循环会一直执行代码，直到条件不再成立。

```C++
while (条件) {
        // 条件成立时反复执行这里的代码
}
```

例如：

```C++
int count = 0;
while (count < 5) {
        cout << "你好，世界！" << endl;
        count++;
}
```

5.1.2 for 循环

`for`循环通常用于知道要执行多少次循环的情况。它由三个部分组成：初始化、条件、更新。

```C++
for (初始化; 条件; 更新) {
        // 条件成立时反复执行这里的代码
}
```

例如：

```cpp
for (int i = 1; i <= 5; i++) {
        cout << i << endl;
}
```

5.1.3 do-while 循环

`do-while`循环和 `while` 循环类似，不同的是它会先执行一次代码，然后再检查条件。

```cpp
do {
        // 先执行这里的代码
} while (条件);
```

例如：

```cpp
int count = 0;
do {
        cout << "你好，世界!" << endl;
        count++;
} while (count < 5);
```

小练习

- 9分钟OJ P1048-1050 分别用for循环，while循环和 do-while 循环实现

5.2 练习

4.3.1 奇数偶数判断

P1051，P1052

4.3.1 整除问题

P1053，P1055，P1056

4.3.2 累加问题

P1061，P1063，P1064

5.3 算法和时间复杂度

5.3.1 什么是算法

算法是一系列指令，用来解决特定问题或完成特定任务。例如，你可以把算法想象成烘焙蛋糕的食谱，其中每个步骤都是一个指令，最终得到一个美味的蛋糕。

5.3.2 什么是时间复杂度

时间复杂度是用来描述一个算法的运行时间如何随着输入数据的大小而变化的。通常用大O符号（O-notation）表示，例如 O(1)、O(n)、O(n^2) 等等。

例子：想象一下，你要在一本书中查找某个单词。如果这本书只有几页，你查找的时间会很短。如果这本书有1000页，那么查找时间会更长。时间复杂度就是用来描述这种情况的。

5.3.3 如何分析时间复杂度

- 首先，找到算法中执行次数最多的那部分操作，通常是一个循环中的某个步骤。例如，在一个求数组和的程序中，基本操作是加法。

```C++
int sum = 0;
for (int i = 0; i < n; i++) {
        // 这个循环执行了n次
        sum += arr[i];
        // 这是基本操作，每次循环都执行
}
```

在这个例子中，加法操作（`sum += arr[i];`）是基本操作。

- 接下来，计算基本操作的执行次数。如果有一个从 0 到 n-1 的循环，那么基本操作执行了 n 次。

```C++
for (int i = 0; i < n; i++) {
        // 这个循环执行了 n 次
}
```

- 在上面的例子中，循环运行了 n 次，因此时间复杂度是 O(n)。

在分析时间复杂度时，我们只关心输入规模非常大的情况，因此我们可以忽略常数项和低阶项。

- **例子1**： $3n+5$ 简化为 $O(n)$
- **例子2**： $n^2+10n+100$简化为 $O(n^2)$