字符串

哈希 & 字典树

邵逸帆 *qωq*

23 电信基地班
兰州大学算法与程序设计集训队

2024 年 7 月 17 日

目录

1. 字符串基础
2. 字符串哈希
3. 字典树

字符类型

- char 类型
- char[] 类型 (char 数组)
- char* 类型 (char 指针)

字符串字面量

eg. `"Hello, World!"`

- type: char[14] in C, const char[14] in C++
- size: 14

Warning: 考虑这样的代码 `char* str = "Hello, World!";` 那么对于字符串的修改会导致未定义行为 (因为你在修改只读内存)。

我们可以使用字符数组来存储字符串，这很直观。

std::string in C++

`std::string` 与 `std::vector<char>` 非常相似，只不过前者提供了更多的字符串操作函数。当然，相较于 `char[]` 更有优势。

std::string

- append() & operator+=: 字符串拼接
- find(): 查找字符串中的某个子串
- substr(): 返回字符串的子串
- operator== & < & > etc.: 字符串比较

诸如 `std::to_string()`,`std::stoi()`,`isdigit()` 等函数有时候会帮助你更优雅地处理字符串。

What is String

字符串是字符的有限序列。对于字符串 s:

- 字符串的长度: 字符串中字符的个数，记为 $|s|$。特别地，空串的长度为 0。
- 子串: $s[i, j]$ 表示从第 i 个字符到第 j 个字符的子串。
- 子序列: $s[i_1, i_2, \cdots, i_k]$ 表示从第 i_1 个字符到第 i_k 个字符的子序列。
- 前缀: $s[1, i]$ 表示从第一个字符到第 i 个字符的前缀。
- 后缀: $s[i, |s|]$ 表示从第 i 个字符到最后一个字符的后缀。
- 回文: s 是回文当且仅当 s 与 s^R 相等，其中 s^R 表示 s 的逆序。
- 字典序: 字符串的字典序是指字符串在字典中的顺序，其中空串是字典序最小的字符串。

字符串的读入

- C style: scanf,getchar,gets(C11 后被弃用)
- C++ style: std::cin,std::getline

对于单一的一个字符，也建议使用字符数组或者 std::string 读入 (因为字符串的读入会跳过空白符)。

输出

- C style: printf,putchar,puts
- C++ style: std::cout,std::endl,std::cerr

std::cerr 是无缓冲的，而 std::cout 是行缓冲的。善用这一特性，对调试有很大帮助。注意：scanf 读入 std::string 时需要提前分配内存。

目录

1 字符串基础

2 字符串哈希

3 字典树

What is Hash

哈希是一种将任意长度的输入通过散列函数变换为固定长度输出的方法。哈希函数的输出通常称为哈希值。

哈希函数

- 一致性: 对于相同的输入，哈希函数应该返回相同的哈希值。
- 高效性: 计算哈希值的时间应该尽可能短。
- 雪崩效应: 输入的微小变化应该导致输出的巨大变化。
- 抗冲突性: 不同的输入应该尽可能产生不同的哈希值。

简单来说，我们让一个非常大的值域映射到一个比较小的值域，尽可能降低冲突的概率，这就是哈希。

简单来说，哈希可以帮助我们快速判断两个数据是否相等。这一思想亦可用于字符串。

我们可以将字符串看做一个数，比如说

$$S = s_1 \times p^{n-1} + s_2 \times p^{n-2} + \cdots + s_n \times p^0$$

其中 p 是一个质数，s_i 是字符串的第 i 个字符 (我们让每一个字符对应一个整数即可)。这样我们就可以用一个数来表示一个字符串。这样的方法有很多好处，比如说我们可以用 $O(1)$ 的时间复杂度来判断两个字符串是否相等。

不过，

目录

1 字符串基础

2 字符串哈希

3 字典树

什么是字典树

pass

THX 4 Listening! :)