


原创 AICodeThunder 已于 2023-11-30 19:18:28 修改 阅读量334 收藏 1 点赞数

分类专栏: C++知识点总结 文章标签: C/C++ 高精度计算 算法

 C++知识点总结 专栏收录该内容

7 订阅 53 篇文章

一、复习高低精度

一个数分为两种类型:

- 1. 高精度数, 即一个长度特别长的数, 使用 long long 也无法存储的一类数字。
- 2. 低精度数, 即一个普通的数, 可以使用 long long 来存储。

由于 高精度除法 比较简单, 建议大家摸透了高精度加减法和高精度乘法的逻辑 (戳蓝色文字进入课程快览)。

二、复习 高精度乘法

- 1. 一共有两个 for 循环, 第一个 for 循环遍历第一个因数, 第二个 for 循环遍历第二个因数。
- 2. for 循环 中的值分别是 $j = 0 \sim \text{len}_b - 1$, $i = 0 \sim \text{len}_a - 1$ 。
- 3. 计算逻辑:

`ans[i+j] = a[i] * b[j] + in + ans[i+j];`

- 4. 输出:

【去前导零】while (ans[len_ans-1] == 0 && len_ans > 1) len_ans--;

【正常逆序输出】ans[len_ans-1] ~ ans[0]

三、存储的基础类型

输入的高精度被除数: char 类型

转换后的高精度被除数: int 类型

低精度除数: int 类型

高精度被除数的位数: int 类型

存储结果: char 类型

答案的长度: int 类型

四、输入与转换

```
1 #include <iostream>
2 #include <cstring>
3 using namespace std;
4
5 int main()
6 {
7     // 存储并输入
8     char a_str[1005] = {};
9     int b;
10    cin >> a_str >> b;
11
12    // 转换
13    int a[1005] = {};
14    int len_a = strlen(a_str);
15    for (int i = 0; i <= len_a-1; i++)
16    {
17        a[i] = a_str[i] - 48; // 正序存储
18    }
19    return 0;
```

 AICodeThunder 关注

0 0 1 0



五、计算过程

```
1  #include <iostream>
2  #include <cstring>
3  using namespace std;
4
5  int main()
6  {
7      // 存储并输入
8      char a_str[1005] = {};
9      int b;
10     cin >> a_str >> b;
11
12     // 转换
13     int a[1005] = {};
14     int len_a = strlen(a_str);
15     for (int i = 0; i <= len_a-1; i++)
16     {
17         a[i] = a_str[i] - 48; // 正序存储
18     }
19
20     // 计算
21     int len_ans = len_a; // 计算次数
22     int ans[1005] = {};
23     int rem = 0; // 余数
24     for (int i = 0; i <= len_ans-1; i++)
25     {
26         ans[i] = (rem * 10 + a[i]) / b; // 写商
27         rem = (rem * 10 + a[i]) % b; // 写余
28     }
29
30     // 去前导零
31     int k = 0; // k 表示第一个不为 0 元素的下标
32     while (ans[k] == 0 && k < len_ans-1)
33     {
34         k++;
35     }
36
37     // 正常正序输出
38     for (int i = k; i <= len_ans-1; i++)
39     {
40         cout << ans[i];
41     }
42     return 0;
43 }
```



```
1  #include <iostream>
2  #include <cstring>
3  using namespace std;
4
5  int main()
6  {
7      // 存储并输入
8      char a_str[1005] = {};
9      int b;
10     cin >> a_str >> b;
11
12     // 转换
13     int a[1005] = {};
14     int len_a = strlen(a_str);
15     for (int i = 0; i <= len_a-1; i++)
16     {
17         a[i] = a_str[i] - 48; // 正序存储
18     }
19
20     // 计算
21     int len_ans = len_a; // 计算次数
22     int ans[1005] = {};
23     int rem = 0; // 余数
24     for (int i = 0; i <= len_ans-1; i++)
25     {
26         ans[i] = (rem * 10 + a[i]) / b; // 写商
27         rem = (rem * 10 + a[i]) % b; // 写余
28     }
29
30     // 去前导零
31     int k = 0; // k 表示第一个不为 0 元素的下标
32     while (ans[k] == 0 && k < len_ans-1)
33     {
34         k++;
35     }
36
37     // 正常正序输出
38     for (int i = k; i <= len_ans-1; i++)
39     {
40         cout << ans[i];
41     }
42     return 0;
43 }
```



C:\Users\秘\Desktop\CSP-J\高精度计算\高精度乘法\高精度除法\高精度除法 (高精度数 ÷ 低精度数).exe

```
39820487382863092
6
6636747897143848
-----
Process exited after 43.05 seconds with return value 0
请按任意键继续. . .
```

六、小数点优化

```
1  #include <iostream>
2  #include <cstring>
3  using namespace std;
4
5  int main()
6  {
7      // 存储并输入
8      char a_str[1005] = {};
9      int b;
10     int point = 3; // 保留小数的数位
11     cin >> a_str >> b;
12
13     // 转换
14     int a[1005] = {};
15     int len_a = strlen(a_str);
16     for (int i = 0; i <= len_a-1; i++)
17     {
18         a[i] = a_str[i] - 48; // 正序存储
19     }
20
21     // 计算
22     int len_ans = len_a; // 计算次数
23     int ans[2100] = {};
24     int rem = 0; // 余数
25     for (int i = 0; i <= len_ans+point-1; i++)
26     {
27         ans[i] = (rem * 10 + a[i]) / b; // 写商
28         rem = (rem * 10 + a[i]) % b; // 写余
29     }
30
31     // 去前导零
32     int k = 0; // k 表示第一个不为 0 元素的下标
33     while (ans[k] == 0 && k < len_ans-1)
34     {
```



```
38 // 正常正序输出整数部分
39 for (int i = k; i <= len_ans-1; i++)
40 {
41     cout << ans[i];
42 }
43
44 // 小数点后判断
45 if (point > 0)
46 {
47     cout << ".";
48     for (int i = len_ans; i <= len_ans+point-1; i++)
49     {
50         cout << ans[i];
51     }
52 }
53 return 0;
54 }
```

A

附录：两数的平均数

```
1 #include <iostream>
2 #include <cstring>
3 using namespace std;
4
5 int main()
6 {
7     // 高精度加法
8     // 输入并存储
9     char a_str[1005] = {};
10    char b_str[1005] = {};
11    cin >> a_str >> b_str;
12    int a[1005] = {};
13    int b[1005] = {};
14    int len_a = strlen(a_str);
15    int len_b = strlen(b_str);
16    for (int i = 0; i <= len_a-1; i++)
17    {
18        a[len_a-i-1] = a_str[i] - 48;
19    }
20    for (int i = 0; i <= len_b-1; i++)
21    {
22        b[len_b-i-1] = b_str[i] - 48;
23    }
24
25    // 计算
26    int sum[1005] = {};
27    int len_sum = max(len_a, len_b);
28    int in = 0;
29    for (int i = 0; i <= len_sum-1; i++)
30    {
31        sum[i] = a[i] + b[i] + in;
32        in = sum[i] / 10;
33        sum[i] %= 10;
34    }
35
36    // 最高位判断
37    if (in)
38    {
39        sum[len_sum] = in;
40        len_sum++;
41    }
42
43
44
45    // 高精度除法
46    int divid[1005] = {};
47    // 正序存储
48    for (int i = 0; i <= len_sum-1; i++)
```



AI Code Thunder

关注

👍 0



🌟 1

💬 0

```

51 }
52
53 // 计算
54 int len_ans = len_sum;
55 int ans[1005] = {};
56 int rem = 0;
57 for (int i = 0; i <= len_ans-1; i++)
58 {
59     ans[i] = (rem * 10 + divid[i]) / 2;
60     rem = (rem * 10 + divid[i]) % 2;
61 }
62
63 // 去前导零
64 int k = 0;
65 while (ans[k] == 0 && k < len_ans-1)
66 {
67     k++;
68 }
69
70 // 输出平均数
71 for (int i = k; i <= len_ans-1; i++)
72 {
73     cout << ans[i];
74 }
75 return 0;
76 }

```

文章知识点与官方知识档案匹配，可进一步学习相关知识

算法技能树 首页 概览 61330 人正在系统学习中

NOIP初赛知识点总结

zsjlziziyang的ti

标题写的是NOIP初赛**知识点总结**，实则是对自己知识疏漏的完善，写这篇博客，希望对读者有所帮助 大部分内容摘自NOIP初赛**知识点**(大全) 计算机发展及应用 1、第一台

【零散知识点总结2】

weixin 44543307的

大部分**知识点**来源于网络，知道的可以在评论区贴上来源喔 内容涵盖：MySQL、Spring、Spring Boot、Spring Cloud、RabbitMQ、Kafka、Linux 等技术栈 零散**知识点**

C++实现高精度除法 高精度除法c++

下面是一个使用C++实现高精度除法的示例代码: #include<iostream>#include<vector>#include<string>using namespace std; int main() { string a, b; int n, m; cin >> a >> b; n = a.size(); m = b.size(); vector<int> ans(n + m, 0); int i = 0, j = 0; while (i < n) { int t = 0; while (j < m) { t = t * 10 + (a[i] - '0') * (b[j] - '0'); j++; } ans[i + j] = t / 9; t = t % 9; i++; j = 0; } for (int k = 0; k < ans.size(); k++) { if (ans[k] != 0) { cout << ans[k] << " "; if (k % 10 == 9) cout << "\n"; } } return 0; }

C++ 的高精度除法 c++高精度除法

对于C++而言,最大的数据为 long long(64b,8位),对于超过 8B 的数据,C++ 没有对应的数据类型进行表示。所以我们需要知道高精度计算。更详细的解释,可以参考这个网页

C++ 的高精度除法 热门推荐

努力中的老周的：

为什么需要高精度计算 对于 C++ 而言，最大的数据为 long long (64b, 8位)，对于超过 8B 的数据，C++ 没有对应的数据类型进行表示。所以我们需要知道高精度计算。

C++高精度除法的实现介绍

zhengddzz的

高精度除法是指在进行除法运算时，处理大数的方法。在C++中，通常使用自定义的数据结构或字符串表示大数，因为内置的数据类型可能无法满足大数运算的需求。与高精度加法类似，高精度除法也需要处理进位和借位的问题。

C++高精度(加减乘除)_c++高精度加法

本条出自C++ vector操作 会敲代码的地质汪的博客-CSDN博客 二、高精度加法 791. 高精度加法 - AcWing题库 模板题: 注意点: 1、大整数的存储 (1)我们将大整数的每一

算法入门系列 C/C++语言的高精度除法(4/4) 高精度除法 c语言-CSDN博...

1.3.怎么实现高精度? 2.高精度的逐步实现 2.1如何接受两个超大的数据? 2.2把 a b数组传入到除法函数 2.3数组加法函数的实现 前言 hello! 各位学习算法的宝子们大家好啊

C/C++ 一些知识点总结

roufoo的时

临时整理的，有些可能不对。1) 如果一个类里面有const-qualifier或reference, compiler不会为它生成default copy assignment operator函数。2) Hidden是指子类 and 父类有

转：高质量C++/C编程指南

shuivingzi5 蝟

高质量C++/C编程指南

文件状态
[]草稿文件
[√]正式文件
[]更改正式文件<

C++高精度除法(超详细,有样例模拟)

与高精度乘法一样,高精度除法也是一个大数字除以一个数。四种高精度运算的思路都很相似。接下来我们以 $175/12 = 14 \dots 7$ 举例讲解。第一步,最初 $r=0$,即余数是零。然后

C++ 基础算法 高精度除法 c++高精度除法

C++ 基础算法 高精度除法 给定两个非负整数(不含前导 0) A,B, 请你计算 A/B 的商和余数。输入格式 共两行, 第一行包含整数 A, 第二行包含整数 B。输出格式 共两行, 第



AI Code Thunder

关注

0

□

★ 1

0