

C++知识点总结(5): 高精度乘法

原创

AICodeThunder

已于 2024-01-06 20:34:05 修改

阅读量1.2k

收藏 3


点赞数 2

分类专栏: C++知识点总结

文章标签: C/C++

编程笔记

知识点总结

 C++知识点总结 专栏收录该内容

7 订阅 53 篇文章

一、高精度数 × 低精度数

1. 输入两个数字

```
1 char a_str[1005] = {};  
2 long long b;  
3 cin >> a_str >> b;
```

2. 将高精度数转换为整型

```
1 int a[1005] = {};  
2 int len_a = strlen(a_str);  
3 for (int i = 0; i <= len_a-1; i++)  
4 {  
5     a[len_a-i-1] = a_str[i] - 48;  
6 }
```

3. 计算

```
1 int len_ans = len_a;  
2 long long ans[1005] = {};  
3 long long in = 0;  
4 for (int i = 0; i <= len_ans-1; i++)  
5 {  
6     ans[i] = a[i] * b + in; // 存储数字  
7     in = ans[i] / 10; // 得到进位  
8     ans[i] %= 10; // 在对应的数位上保留实际得数的最后一位  
9 }
```

4. 输出结果

```
1 while (in > 0) // 最高位处理  
2 {  
3     ans[len_ans] = in % 10;  
4     len_ans++;  
5     in /= 10;  
6 }  
7  
8 // 正常输出  
9 for (int i = len_ans - 1; i >= 0; i--)  
10 {  
11     cout << ans[i];  
12 }
```

5. 注意一个特例先行

```
1 if (a == 0 || b == 0)  
2 {  
3     cout << 0;  
4     return 0;  
5 }
```

6. 完整代码

 AICodeThunder

关注

👍 2

🗨 0

🌟 3

💬 2

```
1  #include <iostream>
2  #include <cstring>
3  using namespace std;
4
5  int main()
6  {
7      // 存储并输出两个数字
8      char a_str[1005] = {};
9      long long b;
10     cin >> a_str >> b;
11
12     // 特例先行: 结果是0的情况
13     if (a == 0 || b == 0)
14     {
15         cout << 0;
16         return 0;
17     }
18
19     // 转换第一个高精度数
20     int a[1005] = {};
21     int len_a = strlen(a_str);
22     for (int i = 0; i <= len_a-1; i++)
23     {
24         a[len_a-i-1] = a_str[i] - 48;
25     }
26
27     // 计算
28     int len_ans = len_a;
29     long long ans[1005] = {};
30     long long in = 0;
31     for (int i = 0; i <= len_ans-1; i++)
32     {
33         ans[i] = a[i] * b + in; // 存储数字
34         in = ans[i] / 10; // 得到进位
35         ans[i] %= 10; // 在对应的数位上保留实际得数的最后一位
36     }
37
38     // 输出结果
39     while (in > 0) // 最高位处理
40     {
41         ans[len_ans] = in % 10;
42         len_ans++;
43         in /= 10;
44     }
45
46     // 正常输出
47     for (int i = len_ans - 1; i >= 0; i--)
48     {
49         cout << ans[i];
50     }
51
52     return 0;
53 }
54 /*
55 注明:
56 由于是从编译器中复制过来的, 所以缩进有些难看, 大家可以自行调整(复制到本地编译器还是可以的)。
57 */
```

二、高精度数 × 高精度数

计算思路改变了一些, 其他不变。

```
1  #include <iostream>
2  #include <cstring>
3  using namespace std;
4
5  int main()
6  {
7      // 存储并输出两个数字
```



AI Code Thunder

关注



2 3 2

```
8 | char a_str[1005] = {}; 9 | char b_str[1005] = {};
10 | cin >> a_str >> b_str;
11 |
12 | // 转换高精度数
13 | int a[1005] = {};
14 | int b[1005] = {};
15 | int len_a = strlen(a_str);
16 | int len_b = strlen(b_str);
17 | for (int i = 0; i <= len_a-1; i++)
18 | {
19 |     a[len_a-i-1] = a_str[i] - 48;
20 | }
21 | for (int i = 0; i <= len_b-1; i++)
22 | {
23 |     b[len_b-i-1] = b_str[i] - 48;
24 | }
25 |
26 | // 计算
27 | int ans[2010] = {};
28 | int in = 0;
29 | for (int j = 0; j <= len_b-1; j++)
30 | {
31 |     for (int i = 0; i <= len_a-1; i++)
32 |     {
33 |         ans[i+j] = a[i] * b[j] + in + ans[i+j];
34 |         in = ans[i+j] / 10;
35 |         ans[i+j] %= 10;
36 |     }
37 |     // 最高位处理
38 |     ans[len_a+j] = in;
39 |     in = 0; // 重置进位
40 | }
41 |
42 | // 正常输出
43 | int len_ans = len_a + len_b; // 结果的最大位数
44 | // 前导0
45 | while (ans[len_ans-1] == 0 && len_ans > 1)
46 | {
47 |     len_ans--;
48 | }
49 |
50 | for (int i = len_ans - 1; i >= 0; i--)
51 | {
52 |     cout << ans[i];
53 | }
54 |
55 | return 0;
56 | }
```

文A

三、高精度平方计算器

首先我们要知道， n 的平方（记作 n^2 ）相当于 $n \times n$ ，其实我们可以按照高精度数 \times 高精度数的思想来完成。

- 想法1：将所有b都改为a。
- 想法2：使用strcpy()函数直接将a的值赋值给b。

建议采用想法2，**示例代码** 如下：

```
1 | #include <iostream>
2 | #include <cstring>
3 | using namespace std;
4 |
5 | int main()
6 | {
7 |     // 存储并输出两个数字
8 |     char a_str[1005] = {};
9 |     char b_str[1005] = {};
10 |    cin >> a_str;
11 |    strcpy(b_str, a_str);
```



AI Code Thunder

关注

👍 2

🗨 0

🌟 3

💬 2

```
12 |
13 |     // 转换高精度数
14 |     int a[1005] = {};
15 |     int b[1005] = {};
16 |     int len_a = strlen(a_str);
17 |     int len_b = strlen(b_str);
18 |     for (int i = 0; i <= len_a-1; i++)
19 |     {
20 |         a[len_a-i-1] = a_str[i] - 48;
21 |     }
22 |     for (int i = 0; i <= len_b-1; i++)
23 |     {
24 |         b[len_b-i-1] = b_str[i] - 48;
25 |     }
26 |
27 |     // 计算
28 |     int ans[2010] = {};
29 |     int in = 0;
30 |     for (int j = 0; j <= len_b-1; j++)
31 |     {
32 |         for (int i = 0; i <= len_a-1; i++)
33 |         {
34 |             ans[i+j] = a[i] * b[j] + in + ans[i+j];
35 |             in = ans[i+j] / 10;
36 |             ans[i+j] %= 10;
37 |         }
38 |         // 最高位处理
39 |         ans[len_a+j] = in;
40 |         in = 0; // 重置进位
41 |     }
42 |
43 |     // 正常输出
44 |     int len_ans = len_a + len_b; // 结果的最大位数
45 |     // 前导0
46 |     while (ans[len_ans-1] == 0 && len_ans > 1)
47 |     {
48 |         len_ans--;
49 |     }
50 |
51 |     for (int i = len_ans - 1; i >= 0; i--)
52 |     {
53 |         cout << ans[i];
54 |     }
55 |
56 |     return 0;
57 | }
```

四、高精度阶乘计算器

注释已经详细地注明了，大家自己看一看即可。

```
1 | #include <iostream>
2 | #include <cstring>
3 | using namespace std;
4 |
5 | int main()
6 | {
7 |     // 输入n的值
8 |     int n;
9 |     cin >> n;
10 |
11 |     // 初始化
12 |     int in = 0;
13 |     int ans[10005] = {1};
14 |     int len_ans = 1;
15 |
16 |     // 计算n!的值
17 |     for (int num = 1; num <= n; num++)
18 |     {
19 |         for (int i = 0; i <= len_ans-1; i++)
```



AI Code Thunder

关注



```
20 |         {21 |             ans[i] = ans[i] * num + in;
22 |             in = ans[i] / 10;
23 |             ans[i] %= 10;
24 |
25 |             // 拓宽长度
26 |             if (i == len_ans-1 && in > 0)
27 |             {
28 |                 len_ans++;
29 |             }
30 |         }
31 |     }
32 |
33 |     // 输出n!的结果
34 |     for (int i = len_ans-1; i >= 0; i--)
35 |     {
36 |         cout << ans[i];
37 |     }
38 |     return 0;
39 | }
```



今日收获

高精度乘法

文件 主页 共享 查看

快速访问

桌面

下载

文档

图片

CSP-J

高精度乘法

高精度计算

音乐

OneDrive

此电脑

网络

名称

修改日期

类型

大小

高精度乘法 (高精度数×低精度数) .cpp2023/11/18 20:15C++ Source File1 KB

高精度乘法 (高精度数×高精度数) .cpp2023/11/18 20:03C++ Source File1 KB

高精度乘法 (阶乘) .cpp2023/11/18 21:01C++ Source File1 KB

高精度乘法 (平方) .cpp2023/11/18 20:55C++ Source File1 KB

4个项目 选中 1 个项目 892 字节

这样，高精度就只剩下下周要研究的 **高精度除法** 啦 (￣▽￣) 加油!

文章知识点与官方知识档案匹配，可进一步学习相关知识

C技能树 首页 概览 206136 人正在系统学习中

C++高精度乘法

C++高精度乘法实现

m0_74004162的博文

c++高精度乘法(五万位*五万位)

c++高精度乘法,字符串做法,支持50000位两数相乘，希望大佬帮助改进。Thanks♪(･ω･)/

2 条评论

Wormwaker

热评

阶乘这段代码不对的，输入1~4都输出0,5输出20

AI Code Thunder

关注

2

3

2