Multiply Strings

Given two numbers represented as strings, return multiplication of the numbers as a string.

Note: The numbers can be arbitrarily large and are non-negative.

Solution 1

This is the standard manual multiplication algorithm. We use two nested for loops, working backward from the end of each input number. We pre-allocate our result and accumulate our partial result in there. One special case to note is when our carry requires us to write to our sum string outside of our for loop.

At the end, we trim any leading zeros, or return 0 if we computed nothing but zeros.

```
string multiply(string num1, string num2) {
    string sum(num1.size() + num2.size(), '0');

    for (int i = num1.size() - 1; 0 <= i; --i) {
        int carry = 0;
        for (int j = num2.size() - 1; 0 <= j; --j) {
            int tmp = (sum[i + j + 1] - '0') + (num1[i] - '0') * (num2[j] - '0')
+ carry;
            sum[i + j + 1] = tmp % 10 + '0';
            carry = tmp / 10;
        }
        sum[i] += carry;
    }

    size_t startpos = sum.find_first_not_of("0");
    if (string::npos != startpos) {
        return sum.substr(startpos);
    }
    return "0";
}
```

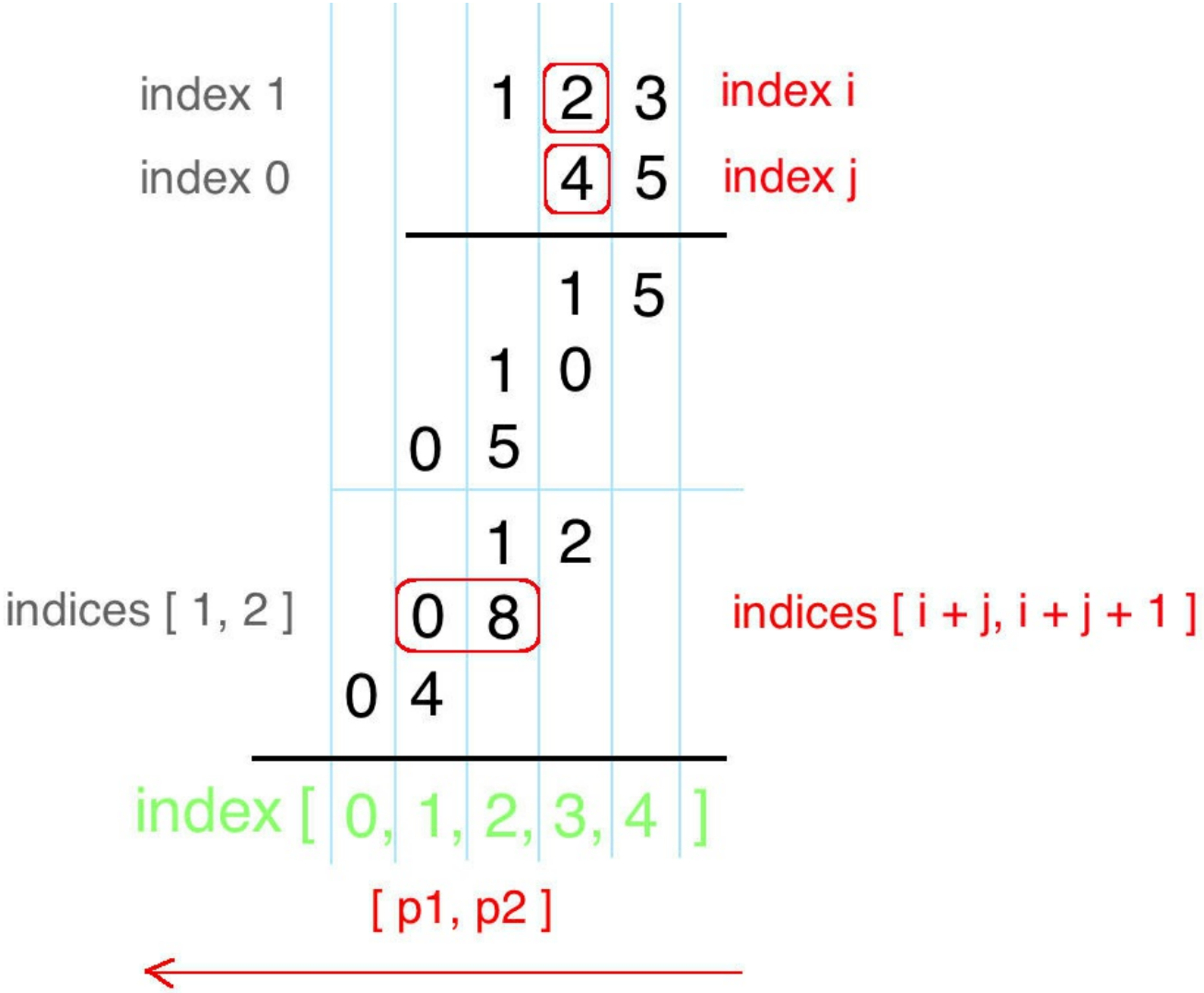
written by ChiangKaiShrek original link here

Solution 2

Remember how we do multiplication?

Start from right to left, perform multiplication on every pair of digits, and add them together. Let's draw the process! From the following draft, we can immediately conclude:

```
`num1[i] * num2[j]` will be placed at indices `[i + j`, `i + j + 1]`
```



Here is my solution. Hope it helps!

```java
public String multiply(String num1, String num2) {
    int m = num1.length(), n = num2.length();
    int[] pos = new int[m + n];

    for(int i = m - 1; i >= 0; i--) {
        for(int j = n - 1; j >= 0; j--) {
            int mul = (num1.charAt(i) - '0') * (num2.charAt(j) - '0');
            int p1 = i + j, p2 = i + j + 1;
            int sum = mul + pos[p2];

            pos[p1] += sum / 10;
            pos[p2] = (sum) % 10;
        }
    }

    StringBuilder sb = new StringBuilder();
    for(int p : pos) if(!(sb.length() == 0 && p == 0)) sb.append(p);
    return sb.length() == 0 ? "0" : sb.toString();
}
```

written by yavinci original link here

Solution 3

```
public class Solution {
    public String multiply(String num1, String num2) {
        int n1 = num1.length(), n2 = num2.length();
        int[] products = new int[n1 + n2];
        for (int i = n1 - 1; i >= 0; i--) {
            for (int j = n2 - 1; j >= 0; j--) {
                int d1 = num1.charAt(i) - '0';
                int d2 = num2.charAt(j) - '0';
                products[i + j + 1] += d1 * d2;
            }
        }
        int carry = 0;
        for (int i = products.length - 1; i >= 0; i--) {
            int tmp = (products[i] + carry) % 10;
            carry = (products[i] + carry) / 10;
            products[i] = tmp;
        }
        StringBuilder sb = new StringBuilder();
        for (int num : products) sb.append(num);
        while (sb.length() != 0 && sb.charAt(0) == '0') sb.deleteCharAt(0);
        return sb.length() == 0 ? "0" : sb.toString();
    }
}
```

If we break it into steps, it will have the following steps. 1. compute products from each pair of digits from num1 and num2. 2. carry each element over. 3. output the solution.

Things to note:

1. The product of two numbers cannot exceed the sum of the two lengths. (e.g. 99 * 99 cannot be five digit)

2.

```
int d1 = num1.charAt(i) - '0';
int d2 = num2.charAt(j) - '0';
products[i + j + 1] += d1 * d2;
```

written by lx223 original link here