Minimum Factorization

Given a positive integer `a`, find the smallest positive integer `b` whose multiplication of each digit equals to `a`.

If there is no answer or the answer is not fit in 32-bit signed integer, then return 0.

Example 1
Input:

```
48
```

Output:
```
68
```

Example 2
Input:

```
15
```

Output:
```
35
```

Solution 1

Reference: http://www.geeksforgeeks.org/find-smallest-number-whose-digits-multiply-given-number-n/

```java
public class Solution {
    public int smallestFactorization(int n) {
        // Case 1: If number is smaller than 10
        if (n < 10) return n;

        // Case 2: Start with 9 and try every possible digit
        List<Integer> res = new ArrayList<>();
        for (int i = 9; i > 1; i--) {
            // If current digit divides n, then store all
            // occurrences of current digit in res
            while (n % i == 0) {
                n = n / i;
                res.add(i);
            }
        }

        // If n could not be broken in form of digits
        if (n != 1) return 0;

        // Get the result from the array in reverse order
        long result = 0;
        for (int i = res.size() - 1; i >= 0; i--) {
            result = result * 10 + res.get(i);
            if (result > Integer.MAX_VALUE) return 0;
        }

        return (int)result;
    }
}
```

written by shawngao original link here

Solution 2

This problem is a little bit tricky. Factors should be selected from 2 to 9 and should be as large as possible.

Any 10 digit solution will overflow.

```
public int smallestFactorization(int a) {
    int k = 9;
    List<Integer> ans = new ArrayList<>();
    if (a <= 9) return a;
    while (a > 1 && k >= 2) {
        if (a % k == 0){
            ans.add(k);
            a = a / k;
        }
        else{
            k--;
        }
    }
    Collections.sort(ans);
    // Integer.MAX_VALUE = 2147483647
    // Note: ans starts at least with 2 (guaranteed to have overflow if the size is
great or equal 10)
    if (a > 10 || ans.size() >= 10)   return 0;
    int num = 0;
    for (int i: ans){
        num *= 10;
        num += i;
    }
    return num;
}
```

written by MichaelPhelps original link here

Solution 3

```
class Solution {
public:
    int smallestFactorization(int a) {
        if (a < 10) return a;
        int j=0;
        int res[40];
        for (int i=9; i>1; i--) {
            while (a%i == 0) {
                a = a/i;
                res[j] = i;
                j++;
            }
        }
        if (a > 10) { // prime factors > 10
            return 0;
        }
        unsigned long long ans = 0;
        for (int i=j-1; i>=0; i--) {
            ans = ans*10 + res[i];
            if (ans > 2147483647) return 0;
        }
        return ans;
    }
};
```

written by sundar.84 original link here