

[网易云C++开发, math]20180328_求被3整除的个数

题目如下:

小Q得到一个神奇的数列: 1, 12, 123,...12345678910,1234567891011...。

并且小Q对于能否被3整除这个性质很感兴趣。

小Q现在希望你能帮他计算一下从数列的第 l 个到第 r 个(包含端点)有多少个数可以被3整除。

输入描述:

输入包括两个整数 l 和 r ($1 \leq l \leq r \leq 1e9$), 表示要求解的区间两端。

输出描述:

输出一个整数, 表示区间内能被3整除的数字个数。

示例1

输入

2 5

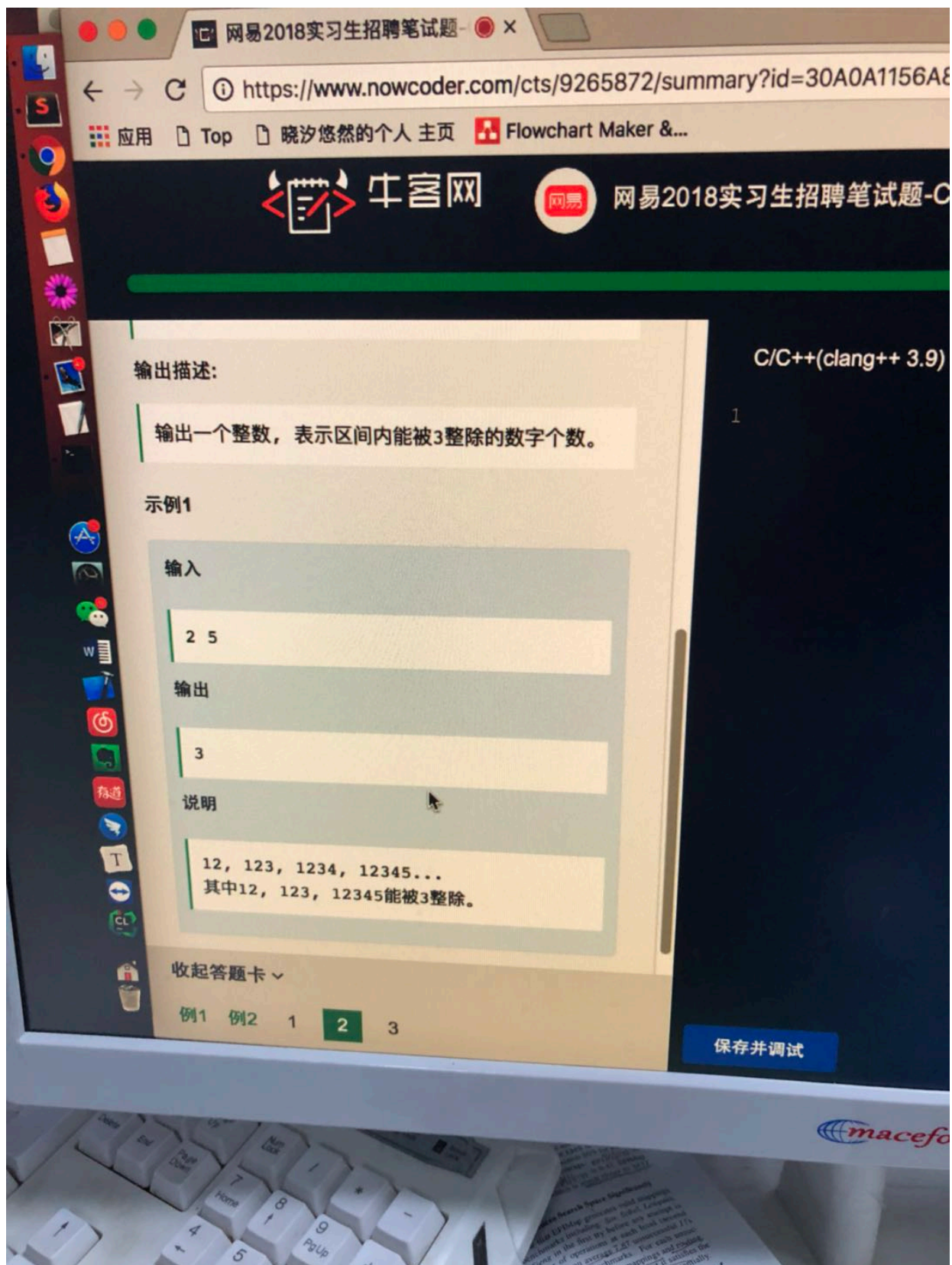
输出

收起答题卡

例1 例2 1 2 3

保存并调试

C/C++(clang++ 3.9)



分析：

第一个数是： 1

第二个数是： 12

第三个数是： 123

第四个数是： 1234

....

第n个数是： 1234 ... n

判断第n个数 能否被3 整除，只要求n的 各个数的和能否被3 整除即可。即 $(1 + 2 + 3 + \dots + n) \% 3 == 0$

也就是 $\{(n+1)n/2\} \% 3 == 0$

```
#include <iostream>
using namespace std;

int div_count(int l, int r) {
    int ret = 0;

    for (int j = l; j <= r; ++j) {
        int k = 1;
        long long sum = ((long long)j + 1) * (long long)j / 2;    /// 对 n
个数字求和
        if (sum % 3 == 0) {
            ret++;
        }
    }
    return ret;
}

int main() {
    int l, r;
    cin >> l >> r;
    cout << div_count(l, r) << endl;
    return 0;
}
```

还有一种方法，找数字规律

```

1  import sys
2  a,b = map(int,sys.stdin.readline().split())
3  i = a
4  j = b
5  count = 0
6  while(i%3!=1):
7      if(i%3 == 0 or i%3 == 2):
8          count += 1
9          i += 1
10 while(j%3!=0):
11     if(j%3 == 2):
12         count += 1
13         j -= 1
14 count += (j-i+1)/3*2
15 print count

```

```

import sys
a,b = map(int,sys.stdin.readline().split())
j = a;
j = b;

count = 0;
while( i % 3 != 1 ):
    if( i % 3 ==0 or i % 3 == 2 ):
        count += 1

    i += 1

while( j % 3 != 0 ):
    if( j%3 == 2 ):
        count += 1

    j -= 1

count += (j-i+1)/3 *2
print count

```