

# ABC 162 解説

kyopro\_friends, gazelle, camypaper, tempura0224, ynymxiaolongbao

2020 年 4 月 12 日

*For International Readers: English editorial will be published in a few days.*

## A:Lucky 7

$N$  を文字列として受け取り、7 を含むかどうかで判定できます。

---

```
1 int main(){
2     char s[4];
3     scanf("%s",s);
4     if(s[0]=='7' || s[1]=='7' || s[2]=='7')puts("Yes");
5     else puts("No");
6 }
```

---

## B:FizzBuzz Sum

実際に各項が数かどうかを判定し、数ならば足すことで答えを求めることができます。

---

```
1 int main(){
2     int n;
3     scanf("%d",&n);
4     long long ans=0;
5     for(int i=1;i<=n;i++){
6         if(i%3!=0 && i%5!=0)ans+=i;
7     }
8     printf("%lld\n",ans);
9 }
```

---

なお、この問題は  $O(1)$  で解くこともできます。

---

```
1 long long sum(long long n){return n*(n+1)/2;}
2 int main(){
3     int n;
4     scanf("%d",&n);
5     long long ans;
6     ans=sum(n)-sum(n/3)*3-sum(n/5)*5+sum(n/15)*15;
7     printf("%lld\n",ans);
8 }
```

---

## C: sum of gcd of tuples (easy)

$\gcd(a, b, c) = \gcd(\gcd(a, b), c)$  が成立します。

$K$  以下の 2 つの数の最大公約数は、ユークリッドの互除法を用いることで  $O(\log K)$  で求めることが出来ます。したがって、実際に全ての  $(a, b, c)$  の組に対して最大公約数を計算することで、この問題は  $O(K^3 \log K)$  で解けました。

C 言語でのユークリッドの互除法の実装例は次のとおりです。

再帰版

---

```
1 int gcd(int p, int q){
2     if(p % q == 0) return q;
3     return gcd(q, p % q);
4 }
```

---

非再帰版

---

```
1 int gcd(int p, int q){
2     while(q != 0){
3         int r = p % q;
4         p = q;
5         q = r;
6     }
7     return p;
8 }
```

---

## D: RGB Triplets

1 つ目の条件を満たす組の数は、 $S$  に含まれる R, G, B の数をそれぞれ  $r, g, b$  としたとき  $rgb$  です。

このうち 2 つ目の条件を満たさない組がいくつあるかを考えます。 $j - i = k - j$  を満たすような組  $(i, j, k)$  の個数は  $O(N^2)$  です。よって、例えば  $i, j$  を固定するといった方法で、この全てを調べて、それが 1 つ目の条件を満たしているかを確認すればいいです。このアルゴリズムの計算量は、後半の全探索がボトルネックになり  $O(N^2)$  です。

## E: sum of gcd of tuples (hard)

各数列  $\{A_i\}$  に対して最大公約数を計算しては間に合いません。そこで、 $1 \leq X \leq K$  に対して「 $\gcd(A_1, \dots, A_N) = X$  となる数列  $\{A_i\}$  がいくつあるか？」という問題を考えます。これが解ければ元の問題にも答えることが出来ます。

最大公約数が  $X$  の倍数であるための必要十分条件は、 $A_1, \dots, A_N$  が全て  $X$  の倍数であることです。そのような数列は  $\lfloor \frac{K}{X} \rfloor^N$  個あります。

ぴったり  $X$  であるための必要十分条件は、「 $X$  の倍数であり、かつ、 $2X, 3X, \dots$  ではない」です。 $X$  が大きい方から順に計算していくことによって、 $2X, 3X, \dots$  の個数を引いて求めることができます。

計算量は  $O(K \log K + K \log N)$  です。

## F:Select Half

この問題は、もし  $N$  が十分小さければ次のような DP で解くことができます。

$DP[i][j] = \{i \text{ 番目までのうちの } 2 \text{ 個も連続しない } j \text{ 個を選んだ時の和の最大値} \}$

この DP には無駄が多いので、ここから状態数を減らします。連続する要素を選んではいけないので、 $i$  番目までの数のうち選べるのは最大で  $\lfloor \frac{i+1}{2} \rfloor$  個です。同様に、残りの  $N-i$  個から選べるのは最大で  $\lfloor \frac{N-i+1}{2} \rfloor$  個なので、最終的に  $\lfloor \frac{N}{2} \rfloor$  個を選ぶためには、 $i$  番目までに  $\lfloor \frac{N}{2} \rfloor - \lfloor \frac{N-i+1}{2} \rfloor = \lfloor \frac{i-1}{2} \rfloor$  個以上選んでいる必要があります。

よって以上より、冒頭の DP で考慮すべき  $j$  の値は、各  $i$  について  $\lfloor \frac{i-1}{2} \rfloor$  以上  $\lfloor \frac{i+1}{2} \rfloor$  以下の高々 3 通りであることがわかるので、状態数及び計算量が  $O(N)$  となり解けました。