I Really **Love** Olympiad Algebra

(25 to 60 pts) Brute Force

With almost no insights, you can already get 60 points by what is basically a brute force.

Let $p(x, y, z) = xyz + 4xy + 2xz + yz + 8x + 4y + 2z$ (i.e. it's the expression given in the problem statement **but without the** k).

Just iterate over all x, y, z such that $p(x, y, z) \leq R$ and then flag all of its different outputs as interesting.

```
vector<bool> interesting(R+1, false);
for(int x = 1; p(x, 1, 1) <= R; x++){
    for(int y = 1; p(x, y, 1) <= R; y++){
        for(int z = 1; p(x, y, z) <= R; z++){
            interesting[p(x, y, z)] = true;
        }
    }
}
```

Note that if any of x, y, or z increases, then $p(x, y, z)$ also increases.

- So, for example, if $p(x, y, z) > R$, then it will also be greater than R for any larger values of z, so we can stop searching.
- If $p(x, y, 1) > R$ (where z is minimal), then it will also be greater than R for any larger values of y or z, so we can stop searching
- Similar logic applies for $p(x, 1, 1)$ and x.

If you iterate over all x, y, and z such that **each one** is less than or equal to some fixed upper bound, then you can get 25 to 45 points. But if you do something similar to the code snippet above, where e.g. the value of x affects the bounds for y and z, then you can get 60 points!

Despite the three nested for loops, we claim that this actually runs in $\mathcal{O}(R \ln^2 R)$. The argument is quite fun, and they key is that we **don't** use a fixed bound for all three loops; rather, the values of the outer loops are used to further restrict the range of values visited by the inner loops. Our argument boils down to the fact that $p(x, y, z) \approx xyz$, so our analysis will be equivalent to enumerating all triples (x, y, z) such that $xyz \leq R$.

We invite you to ask for more details in the Discord if you're curious!

(80 to 93 points)

Note that the given polynomial can be factored into this expression:

$$(x + 1)(y + 2)(z + 4) - 8 + k$$

Noting that k and 8 are constants, we rephrase our problem as follows.

We say that a number n is interesting if there exist positive integer x, y, and z such that $n = (x+1)(y+2)(z+4)$. Then, we wish to count the interesting numbers from $L - (k-8)$ to $R - (k-8)$.

Consider this informal motivation. Suppose for some number n, we can write it as $n = abc$. Then, n is interesting because we can choose $x = a - 1$ and $y = b - 2$ and $z = c - 4$. Therefore it seems that most numbers are interesting, with a few exceptions that might make this argument fall apart.

- n doesn't have enough factors, so it can't be written as $n = abc$
- Even if n does have enough factors, they might not all be large enough (because note that $a - 1$ and $b - 2$ and $c - 4$ must all be **positive**)

Claim. A positive integer n is **not** interesting if any of the following apply.

- $n < 30$
- n is a prime
- n is a product of two primes
- n is 4 multiplied by a prime.
- $n = 32$

Proof. As a reminder, we wish to find three integers a, b, and c such that,

- $n = abc$
- $a \geq 2$
- $b \geq 3$
- $c \geq 5$

The proof is by case bash. Consider the number of prime factors of n.

Case 1: 1 prime factor

If n is prime, then only one of a, b, c can be greater than 1. So, always impossible.

Case 2: 2 prime factors

If n has two prime factors, then only two of a, b, c can be greater than 1. So, always impossible.

Case 3: 3 prime factors

Suppose $n = p_1 \times p_2 \times p_3$ where without loss of generality, assume $2 \leq p_1 \leq p_2 \leq p_3$ and these values are all prime.

Now, let's case bash on the value of p_2, the "middle" value.

If $p_2 = 2$, then the task is impossible, because both b and c must be ≥ 3, but here, only p_3 is possibly ≥ 3. Note that $p_1 = 2$ is forced.

If $p_2 = 3$, then the task is possible if and only $p_3 \geq 5$; we need at least one value ≥ 5 (for c), and if $p_3 \geq 5$, then we can assign $a = p_1$ and $b = p_2$ and $c = p_3$. Note that p_1 can only be 2 or 3.

If $p_2 \geq 5$, then $p_3 \geq p_2 \geq 5$, and so the task is always possible, by assigning $a = p_1$ and

$b = p_2$ and $c = p_3$.

Thus, the following cases are impossible:

- $n = 2 \times 2 \times \text{prime}$
- $n = 2 \times 3 \times 3$
- $n = 3 \times 3 \times 3$

Case 4: 4 prime factors

Suppose $n = p_1 \times p_2 \times p_3 \times p_4$ where without loss of generality, assume $2 \le p_1 \le p_2 \le p_3 \le p_4$ and these values are all prime.

Note that $p_2 \times p_3 \ge 3$, so we can always assign $a = p_1$ and $b = p_2 \times p_3$. If $p_4 \ge 5$, then the task is possible because we can assign $c = p_4$.

Otherwise, suppose that $p_4 < 5$. Then, there are only few cases we need to check. For each one, we can prove or disprove their interestingness by hand since there are only so many ways to distribute the prime factors.

- $2 \times 2 \times 2 \times 2$ (impossible)
- $2 \times 2 \times 2 \times 3$ (impossible)
- $2 \times 2 \times 3 \times 3$ (possible: $a = 2$, $b = 3$, $c = 2 \times 3$)
- $2 \times 3 \times 3 \times 3$ (possible: $a = 2$, $b = 3$, $c = 3 \times 3$)
- $3 \times 3 \times 3 \times 3$ (possible: $a = 3$, $b = 3$, $c = 3 \times 3$)

Thus, only the following cases are impossible:

- $n = 2 \times 2 \times 2 \times 2$
- $n = 2 \times 2 \times 2 \times 3$

Case 5: 5 prime factors.

Suppose $n = p_1 \times p_2 \times p_3 \times p_4 \times p_5$ where without loss of generality, assume $2 \le p_1 \le p_2 \le p_3 \le p_4 \le p_5$ and these values are all prime.

If $p_5 \ge 3$, then $p_4 \times p_5 \ge 5$, so we can always assign $a = p_1$ and $b = p_2 \times p_3$, and $c = p_4 \times p_5$.

Otherwise, suppose that $p_5 < 3$. This actually only leaves us with one case:

- $n = 2 \times 2 \times 2 \times 2 \times 2$

which we can show by hand to be impossible.

Case 6: 6 or more prime factors.

Suppose $n = p_1 \times p_2 \times p_3 \times p_4 \times p_5 \times q$ where without loss of generality, assume $2 \le p_1 \le p_2 \le p_3 \le p_4 \le p_5$ and these values are all prime, and also $q \ge 2$.

Then, the task is always possible by assigning $a = p_1$, $b = p_2 \times p_3$, and $c = p_4 \times p_5 \times q$.

In summary

Here are all the impossible cases:

- prime
- $\text{prime} \times \text{prime}$

- $4 \times \text{prime}$
- 18
- 27
- 16
- 24
- 32

But note that if $a \geq 2$ and $b \geq 3$ and $c \geq 5$, then $n = abc \geq 30$, so we can clean this up a bit:

- $\text{less than } 30$
- prime
- $\text{prime} \times \text{prime}$
- $4 \times \text{prime}$
- 32

leaving only 32 as a special exception.

For $R \leq 5 \times 10^7$, we can find all numbers that satisfy any of the above criteria by using a modified Sieve of Eratosthenes, which will run in $\mathcal{O}(R \ln \ln R)$.

For the subtask where $R \leq 10^8$, you either need to be have a good implementation with low constant factors, *or* you can use a linear sieve instead.

(100 pts) Number Theory Black Magic

There is a somewhat-standard technique for computing the sums of number theoretic functions in sublinear time.

It involves abusing the following two properties of $\text{floor}(n/k)$:

- If n is fixed, then this expression only evaluates to at most $2\sqrt{n}$ different values as k varies, half of which are "big" and half of which are "small".
- $\text{floor}\left(\dfrac{\text{floor}\left(\dfrac{n}{a}\right)}{b}\right) = \text{floor}\left(\dfrac{n}{ab}\right)$

You can prove both of these from the Division Algorithm (i.e. the $n = pq + r$ theorem). These two facts allow us to create a crazy DP with only $\mathcal{O}(\sqrt{n})$ states, which allows us to compute the sum in sublinear time.

For the actual details, you can read the details in this blog: https://codeforces.com/blog/entry/91632 It exactly details the algorithm for counting the number of primes less than or equal to some n. You can then use this function as a blackbox for counting the number of semiprimes and squares of primes.

A running time of $\mathcal{O}(n^{3/4})$ gets you 99 points. To get 100 points, you need to either optimize your implementation to have a low constant factor, *or* implement the $\widetilde{\mathcal{O}}(n^{2/3})$ algorithm described in the blog.