Misket Çözüm

Her $m_l \leq m \leq m_r$ ve $t_l \leq t \leq t_r$ için üzülen torun sayılarını tablo yaparsak bir örüntü görebiliriz. Her sayı sağa gittikçe genelde 1 azalmaktadır, 0'da sabit kalmaktadır veya yeni bir değere güncellenmektedir. 0'da sabit kalma durumunu elemek için 0'da sabit kalmak yerine negatif sayılara indirirsek iki durum kalmış olur. 1 azalma durumunu da elemek için bütün sayılara soldan ne kadar uzaksa o sayı eklenir. Böylece sayılar sağa giderken çoğu zaman sabit kalır, bazen güncellenir. Güncellendiği sayılar t'nin $m - c - 1$'i böldüğü yere denk gelmektedir. Bu t değerlerini başta sieve ile buluruz, daha sonra sağa doğru giderken bütün t değerlerini aynı anda işleyebiliriz.

10 12 80 100 5 girdisi için örnek tablo:

0	-1	-2	-3	-4	-5	4	3	2	1	0	-1	-2	-3	-4	-5	4	3	2	1
-3	-4	-5	5	4	3	2	1	0	-1	-2	-3	-4	-5	5	4	3	2	1	0
4	3	2	1	0	-1	-2	-3	-4	-5	6	5	4	3	2	1	0	-1	-2	-3

Soldan uzaklığını ekleyince:

0	0	0	0	0	0	10	10	10	10	10	10	10	10	10	10	20	20	20	20
-3	-3	-3	8	8	8	8	8	8	8	8	8	8	8	19	19	19	19	19	19
4	4	4	4	4	4	4	4	4	4	16	16	16	16	16	16	16	16	16	16

100 Puan Alan Kod

```cpp
#include <bits/stdc++.h>
#define inf 1000000000
using namespace std;
int tl, tr, ml, mr, c, D;

int uzulen(int m, int t) {
    if (m % t <= c && m >= t) return 0;
    else return t - m%t;
}

int main() {

    cin >> D;
    while (D--) {
        cin >> tl >> tr >> ml >> mr >> c;

        vector<int> en_buyuk_bolen(mr+1, 0);
        for (int i = tl; i <= tr; i++) {
            for (int j = i; j <= mr; j += i) {
                en_buyuk_bolen[j] = i;
            }
        }

        int en_kotu = 0;
        for (int t = tl; t <= tr; t++) {
            en_kotu = max(en_kotu, uzulen(ml, t));
        }

        int cevap = ml;
        int cevap_en_kotu = en_kotu;
        for (int m = ml; m <= mr; m++) {

            if (m-c-1 > 0 && en_buyuk_bolen[m-c-1] != 0) {
                en_kotu = max(en_kotu,
                uzulen(m, en_buyuk_bolen[m-c-1]) + (m-ml));
            }

            if (en_kotu - (m-ml) <= cevap_en_kotu) {
                cevap_en_kotu = max(0, en_kotu - (m-ml));
                cevap = m;
            }
        }
        cout << cevap << '\n';
    }
}
```