Назив проблема: Царина

|  |  |
| --- | --- |
| Аутор: Петар Величковић | Анализа: Петар Величковић |

|  |  |
| --- | --- |
| Тагови: | математика, грамзиви алгоритми, ад-хок |

## Решење и анализа:

Задатак Царина спада у проблеме средње тежине на другим Квалификацијама, и одликује се изузетно кратким решењем; идеја је да решаваоци много више времена потроше на смишљање правилног решења него на саму имплементацију.

Пре свега, приметимо да је једини ефекат броја ограничавање колико игрица можемо изнети из места А; ово је еквивалентно томе да је . Пошто је број ограничен са , ово само по себи треба да сугерише да је тражени алгоритам реда величине ; међутим да бисмо дошли до овог алгоритма неопходно је најпре направити неколико запажања.

Најпре приметимо да је једна од оптималних стратегија врло једноставна: сваки пут пунимо шлепер до врха, и доносимо игрице на први следећи пункт, где губимо једну и истоварамо остале; ово понављамо док не пренесемо све игрице на први пункт. Овим смо изгубили игрица, тј. најмањи цео број већи или једнак . Ако овако наставимо, пренели смо максималан могућ број игрица у место Б; ово можемо наивно представити следећим C++ кодом:

long long carina(long long n, long long c, long long l)

{

for (long long i=0;i<l;i++)

{

long long lost = n/c;

if (c \* lost != n) lost++;

n -= lost;

}

return n;

}

Ово решење је очигледно сложености , и доноси ~50 бодова. Запажање које нам помаже да дођемо до бржег решења је да приметимо да се више корака могу ”спаковати” у један, тј. често ће постојати више од једног корака током ове стратегије где ћемо изгубити исти број игрица; ако ефикасно одредимо број корака који смемо направити тако да у сваком кораку губимо исти број игрица, онда можемо шлепер одмах носити до те позиције; тј. не морамо прелазити један по један пункт.

Да бисмо одредили ову позицију, потребно је посматрати остатак при дељењу са . Назовимо овај остатак . Неопходан број повратака на пункт са којег смо кренули тренутни корак се смањује за бар један када се овај остатак изгуби; како у сваком јединичном кораку губимо игрица, број потребних корака да се ово деси је . Уколико је , онда можемо узети .

Након што одредимо број потребних корака, можемо одмах отићи са пункта на пункт и успут одузети од , наравно водећи рачуна о томе да ли смо прекорачили место Б.

Као резултат добијамо следећи кратак C++ код:

long long carina(long long n, long long c, long long l)

{

while (l > 0 && n > 0)

{

long long D = (n/c) + (n % c > 0);

long long R = (n % c);

if (R == 0) R = c;

long long K = (R / D) + (R % D > 0);

if (K > l) K = l;

n -= K\*D;

l -= K;

}

return max(0LL, n);

}

Приметимо да ово решење у сваком кораку смањује број ”повратака” шлепера за бар један, дакле сложеност овог алгоритма је , што смо на почетку ограничили са Овај алгоритам доноси 100 бодова.