Проблем 2. Степен

За два дата природна броја a и b ($2 \leq a \leq b < 2^{64}$) одредити природне бројеве x и k тако да важи $a \leq x^k \leq b$, а да је k што веће могуће. У случају да има више решења исписати оно у којем је x најмање.

Улаз. (Улазни подаци се учитавају на стандардног улаза) У првој и јединој линији стандардног улаза се налазе природни бројеви a и b.

Излаз. (Излазне подаци се исписују на стандардни излаз) На стандардни излаз исписати редом два тражена природна броја x и k.

Пример 1.

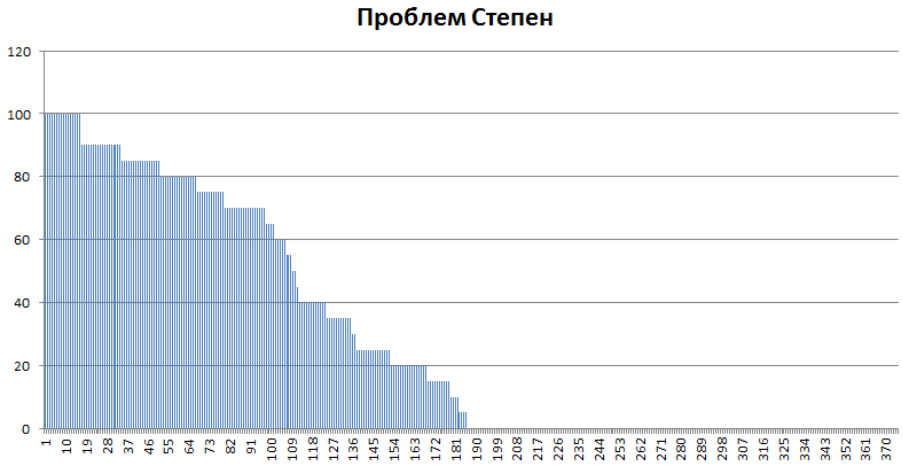
stepen.in	stepen.out
5 20	2 4

Пример 1.

stepen.in	stepen.out
35 50	6 2

Ограничења. Временско ограничење: $0.5s$; Меморијско ограничење: $16B$

Решење и анализа. Други проблем на квалификацијама био је математичке природе. Решење проблема се врло једноставно може исказати у виду математичког алгоритма. Међутим како се ми овде играмо у бинаром и ограниченом свету, није све тако лако као што изгледа. Просечан број бодова у току такмичења био је око 28, док је 50% такмичара освојило позитиван број бодова на њему. Само шеснаест такмичара је имало максимални учинак.



Слика 1. График броја освојених бодова свих такмичара.

Прва ствар коју треба приметити јесте да су могуће вредности степена k јако дискретне.

Наиме, како су границе сегмента ограничене са 2^{64}, а имајући у обзир чињеницу да је x природан број, имамо да тражени степен k не може бити већи од 63. Дакле, k узима вредности из скупа $\{1, 2, \ldots, 63\}$.

У тексту проблема се захтева минимизација број x за тражено k. Препоставимо да смо степен k фиксирали. Које су могуће вредности за број x? Уколико дати услов $a \leq x^k \leq b$ мало трансформишемо, добијамо да важи

$$\sqrt[k]{a} \leq x \leq \sqrt[k]{b}$$

Како је x природан број, минимална вредност која задовољава горње неједнакости може имати два облика:

$$\lfloor \sqrt[k]{a} \rfloor \qquad \text{или} \qquad \lfloor \sqrt[k]{a} \rfloor + 1$$

Једноставним испитивањем да ли прва односно друга могућност задовољава и горњу границу, налазимо тражено x за дато k. Како k може имати само 63 могућих вредности, испитивањем сваке могућности долазимо до коначног решења. Овде треба напоменути да случај $k = 1$ треба посебно да се издвоји. Пошто се тражи максимално k, можемо кренути од максималне могуће вреднити, односно 63, и редом испитивати за вредност x. Чим наиђемо на прво k за које постоји x које задовољава услове, програм може прекинути рад и вратити ове вредности.

Алгоритам: Псеудо код алгоритма проблема Степен

Input: крајеви сегмента a и b
Output: тражене вредности x и k

for $k \leftarrow 63$ **to** 2 **do**
  $x = \lfloor \sqrt[k]{a} \rfloor$;
  if $(x^k < a)$ **then**
    $x = x + 1$;
  end
  if $(a \leq x^k \leq b)$ **then**
    return (x, k)
  end
end
return $(a, 1)$

Дата ограничења захтевају рад са 64-битним типовима података. У *C*-у можемо користити тип *unsigned long long*, а у *Pascal*-у тип *Qword*. Оба типа узимају вредности из сегмента $[0, 2^{64} - 1]$, што је управо нама и потребно. За рачунање вредности x, односно k-тог корена броја a, користимо функције:

- *pow (a, 1/k)* у *C*-у
- *Exp(ln(a) / k)* у *Pascal*-у (ово је заправо "вештачка" верзија функције степена јер користимо природни логаритам као међукорак, односно $\sqrt[k]{a} = a^{1/k} = e^{\ln(a^{1/k})} = e^{\frac{1}{k} \ln a}$)

Међутим као што смо напоменули, овде имамо један мали проблем. Наиме, при позиву фунцкције *pow* која као параметар прима тип *double* може, за веће вредности улазних података, изгубити на тачности. Примера ради извршавањем следећег кода

```
unsigned long long a = 18446744073709551615uLL;
double tmp = (double) a;
printf ("\%lf \n", a);
```

на стандардном излазу ће бити исписано: 184467440737095520000.000000. Међутим, ову замку смо избегли, а да тога нисмо ни свени. Наиме, у коду проверавамо да ли је $pow(x, k) < a$ и ако јесте повећавамо x за један. Овим додатним испитивањем сигурно добијамо тражену вредност x као цео број (наравно ово не важи за реалну вредност). Велики број такмичара је због овог превида, освојио 85 бода.

Сложеност овог алгоритма је тешко тачно израчунати. Међутим, она је, могло би се рећи константна, јер у сваком од 63 корака имамо рачунање степена који може бити имплементиран у линеарном или (мада овде непотребно) логаритамском времену. У овом проблему је акценат био на прецизности, а не на брзини.

Напомена. Друга могућност је била импелементација посебне функције за рачунање k-тог корена. На овај начин пребацивање у *double* не би било потребно, тако да се не би губила горе описана презицност. Наведена функција се може имплементирати уз помоћ бинарне претраге.