# Identična jaja (jaje)

Zamislite da imate $k$ savršeno jednakih jaja - jednaka su u tome što ako ih bacite sa neke iste visine uvijek će se ili sva razbiti ili će sva ostati čitava. Naravno, ako kada se bace sa neke visine ostaju čitava, onda ostaju čitava i kada se bace sa bilo koje manje visine; i obrnuto - ako kada se bace sa neke visine jaja razbiju, onda će se razbiti i ako se bace sa bilo koje veće visine. Također, svaki put kada ih bacite i ona ostanu čitava, ni na koji način se ne oštete niti se promjene. Postavlja se pitanje sa koje najveće visine se ova jaja mogu baciti a da ona ostanu čitava. Preciznije, ako imamo neku zgradu visine $n$ spratova, treba naći najviši sprat $m$ u ovoj zgradi sa kojeg kad se jaje baci, ostaje čitavo. Spratovi su numerisani brojevima od 0 do $n-1$. Nikada se neće desiti da se jaje razbije kada se baci sa sprata 0. Kako bi igra bila zanimljivija, cilj je otkriti sprat $m$ u što manje koraka - sjetite se da imate tačno $k$ identičnih jaja!

## Zadatak

Zadatak je napisati program koji će u što je moguće manje bacanja odrediti sa kojeg se najvišeg sprata mogu baciti jaja a da ostanu čitava. Na početku izvršavanja se poziva funkcija $PronadjiSprat(n, k)$ koju vi implementirate, a čime se daje informacija da je zgrada sa koje se jaja bacaju visoka $n$ spratova i da imate tačno $k$ jednakih jaja. $n$ i $k$ su cijeli brojevi. Specijalno, $k = -1$ znači da imate neograničen broj jaja. Vaša funkcija $PronadjiSprat$ treba da poziva funkciju $BaciJaje(i)$ koja je implementirana od strane komisije, a koja vraća logičku vrijednost tačno ako se jaje razbije kada se baci sa $i$-tog sprata, i suprotno, vrijednost netačno ako jaje ostaje čitavo. Vodite računa da ako pozovete funkciju $BaciJaje(i)$ i ona vrati logičku vrijednost tačno, onda je jedno od vaših početnih $k$ jaja razbijeno i više ga nemate! Cilj je odrediti broj $m$ u što je moguće manje poziva funkcije $BaciJaje$. Vaša funkcija $PronadjiSprat$ treba da vrati ovaj broj $m$.

### Bodovanje

Za svaki testni slučaj, komisija definiše broj $t_{max}$ koji predstavlja optimalan broj bacanja (poziva funkcije $BaciJaje$) sa kojim se može pogoditi sprat $m$ u najgorem slučaju. Broj $t_{max}$ za pojedine testne slučajeve nije javan jer otkriva način rješavanja zadatka, ali je garantovano takav da se sprat $m$ može odrediti unutar $t_{max}$ bacanja. Ukoliko vaša implementacija za neki testni slučaj unutar jednog podzadatka ispravno odredi broj $m$ ali u $t$ pokušaja za koji je $t > t_{max}$, onda dobijate $(50\%)^{\frac{t-t_{max}}{t_{max}}}$ poena za podzadatak kojem ovaj testni slučaj pripada. Ukoliko se slično desi na još nekom testnom primjeru unutar istog podzadatka, postotak poena koje dobijate se množi sa postocima poena za ostale testne slučajeve. Ukoliko vaša implementacija ne odredi ispravno broj $m$ za bilo koji testni slučaj unutar jednog od podzadataka, gubite sve poene na taj podzadatak. *Na primjer.* Ako je za jedan testni slučaj $t_{max} = 150$ ustanovljeni optimalan broj bacanja unutar kojeg se može odrediti broj $m$ u najgorem slučaju, a vaša implementacija ispravno odredi broj $m$ ali u $t = 217$ bacanja, onda prema gornjoj formuli dobijate $\approx 73,37\%$ poena za ovaj podzadatak. Ako se na nekom drugom testnom slučaju u istom podzadatku odredi broj $m$ ali u $t = 250$ bacanja a vrijedi $t_{max} = 100$, onda za ovaj podzadatak dobijate $\approx 73,37\% \cdot 35,36\% \approx 25,94\%$ bodova.

### Primjer

Na početku izvršavanja programa, poziva se vaša funkcija $PronadjiSprat(15, 1)$. Ova funkcija poziva funkciju $BaciJaje$ (koja je implementirana od strane komisije) potreban broj puta:

$BaciJaje(1) = netačno$

$BaciJaje(2) = netačno$

$BaciJaje(3) = netačno$

...

$BaciJaje(11) = netačno$

$BaciJaje(12) = tačno$

Sada je jasno da je traženi sprat $m = 11$ obzirom da kada se jaje baci sa 11. sprata, ono ostaje čitavo; a kada se baci 12. sprata, razbije se. Primijetimo da je bilo potrebno $t = 12$ bacanja jajeta da bi se došlo do ovog zaključka. U ovom testnom primjeru, definisano je $t_{max} = 14$ obzirom da je to broj potrebnih bacanja da se otkrije sprat $m$ u najgorem slučaju, pa bi za ovakvo izvršavanja na ovom testnom primjeru bili dodijeljeni svi bodovi.

**Podzadatak 1 (4 boda):** $k = 1$, $1 \leq n \leq 10.000$

**Podzadatak 2 (7 bodova)**

$k = -1$ (Odnosno, za sve testne slučajeve u ovom podzadatku imate neograničen broj jaja.)
$1 \leq n \leq 100.000$

**Podzadatak 3 (30 bodova):** $k = 2$, $1 \leq n \leq 100.000$

**Podzadatak 4 (59 bodova):** $3 \leq k \leq 1.000$, $1 \leq n \leq 10.000.000$

## Detalji implementacije

Sa servera za takmičenje možete preuzeti pripremljena okruženja (`jaje_c.zip`, `jaje_cpp.zip` ili `jaje_pas.zip`) sa osnovnim fajlovima za C/C++ i Pascal.

Ukoliko koristite C ili C++ napišite funkciju sa prototipom

```
int PronadjiSprat(int n, int k);
```

u fajlu `jaje.[c/cpp]`.

Ukoliko koristite Pascal napišite funkciju sa prototipom

```
function PronadjiSprat(n : LongInt; k : LongInt) : LongInt;
```

u fajlu `jaje.pas`.

Samo unutar ovog fajla treba da implementirate svoje rješenje. Pri tome smijete koristiti i druge pomoćne funkcije koje ste vi napisali, te standardna zaglavlja/biblioteke odabranog programskog jezika i funkcije iz ovih biblioteka. Ne smijete ni na koji način vršiti interakciju sa standardnim ulazom/izlazom niti sa bilo kojom datotekom.

Procedura `PronadjiSprat` može i treba da poziva funkciju `BaciJaje` sa prototipom

```
bool BaciJaje(int i);
```

ukoliko koristite C ili C++, odnosno funkciju sa prototipom

```
function BaciJaje(i : LongInt) : Boolean;
```

ukoliko koristite Pascal.

Ova funkcija i ostala funkcionalnost koja simulira ponašanje prilikom testiranja rješenja na serveru, implementirana je u fajlu `stub.[c/cpp]` ako koristite C/C++, odnosnu u fajlovima `jajelib.pas` i `stub.pas` ako koristite Pascal, a koji se nalaze u pripremljenom okruženju. U ovim fajlovima implementirana je sva interakcija potrebna za testiranje: poziv funkcije `PronadjiSprat` sa parametrima $n$ i $k$, te ispravna povratna vrijednost na pozive funkcije `BaciJaje`. Kada šaljete svoje rješenje, šaljete samo fajl `jaje.[c/cpp/pas]`, dok komisija koristi svoj `stub.[c/cpp/pas]` i eventualno `jajelib.pas` koji nisu javni. U skladu s tim, slobodni ste da modifikujete `stub.[c/cpp/pas]` i po potrebi `jajelib.pas` i prilagođavate ih svojim potrebama u svrhu testiranja na lokalnom računaru.

Ukoliko koristite *Code::Blocks*, u pripremljenim okruženjima možete naći i odgovarajuće projekte sa podešenim parametrima za prevođenje. "*Release build*" u potpunosti odgovara parametrima za prevođenje koji su na serveru za takmičenje, dok "*Debug build*" ima isključene optimizacije i uključene simbole za debagiranje.

Ukoliko koristite *FreePascal IDE*, dovoljno je da pokrenete prevođenja fajla `stub.pas` dok su u istom folderu fajlovi `jaje.pas` i `jajelib.pas`. Na serveru za takmičenje postavljeni su sljedeći parametri za prevođenje: `-dEVAL -vw -XS -O2`.

Ukoliko ne koristite *Code::Blocks*, odnosno *FreePascal IDE*, u okruženjima se nalaze i fajlovi `prevedi_[c/cpp/pas].sh` koje možete koristiti za prevođenje svojih programe, a koje pozivate iz terminala komandom `sh prevedi_[c/cpp/pas].sh` iz odgovarajućeg foldera.