Obstaja polje N bitov – $a_0,\ldots,a_{N-1}$. V eni poizvedbi lahko izbereš podmnožico mest v zaporedju in obrneš bite na teh mestih. Obrat spremeni bit 0 v 1 in bit 1 v 0. Po vsaki poizvedbi dobiš dolžino najdaljšega podpolja, sestavljenega iz samih enic. **Poizvedbe so obstojne – tj. obrnjen bit v poizvedbi, ostane obrnjen do naslednjega obrata.**

Iščeš najdaljše podpolje, sestavljeno zgolj iz enic po opravljenih poizvedbah. Napiši program, ki bo to storil v karseda malo poizvedbah.

Opombe k implementaciji

Vsak test sestoji iz T podtestov.

Tvoj program naj implementira funkcijo

```
std::pair<int,int> find_longest_subarray_of_ones(int n);
```

Poklicana bo T-krat in sprejme celo število N kot parameter – dolžina skritega polja. Funkcija naj vrne `pair{l,r}`, kjer je `l` kazalec na skrajni levi del podpolja in je `r` kazalec na skrajni desni del podpolja.

Poizvedbo opraviš tako, da pokličeš funkcijo `flip_bits`.

```
int flip_bits(const std::vector<bool> &flips);
```

Funkcija prejme vektor N bitov $flips_0,\ldots,flips_{n-1}$, kjer $flips_i=1$ označuje, da je treba obrniti a_i. Po obratu želenih bitov funkcija vrne dolžino najdaljšega podpolja enic v zaporedju.

Oddaj datoteko `ones.cpp` na sistem. V datoteki naj bo implementirana funkcija `find_longest_subarray_of_ones`. Vsebuje lahko tudi dodatno kodo in funkcije, ki jih potrebuješ za rešitev problema, ne sme pa vsebovati funkcije `main`. Pravtako ne beri iz standardnega vhoda in ne piši na standardni izhod. Program naj vključi vzglavno datoteko `ones.h` kot ukaz predprocesorju:

```
#include "ones.h"
```

Lokalno preizkušanje

Podana je datoteka Lgrader.cpp, ki jo lahko prevedeš skupaj s svojim programom za preizkušanje. Program bo prebral število podtestov T v testnem primeru. Sledi mu opis vsakega podtesta. Za vsak podtest je najprej podan N, ki mu sledi $a_0\ldots a_{N-1}$. Če se program izvede pravilno, izpiše 1 in število poizvedb, ki jih je program napravil, drugače izpiše 0.

Omejitve

$T=5$

$1\le N\le 10^4$

$0\le a_i\le 1$

Točkovanje

Število točk, ki jih prejmeš, je odvisno od maksimalnega števila poizvedb, ki jih je napravil tvoj program, da je uspešno rešil podtest. Rezultat se izračuna po formuli:

$$\left(\frac{85}{max\ (85, Q)}\right)^{0.294} \times 100$$

Primer komunikacije

Naj bo začetno polje 1, 0, 1. Ena izmed komunikacij se glasi:

Tekmovalčeva funkcija	Sodniški program
	Pokliče **find_longest_subarray_of_ones(3)**
Pokliče **flip_bits({0, 0, 0})**	Vrne vrednost **1**
Pokliče **flip_bits({0, 1, 0})**	Vrne vrednost **3**
Vrne vrednost **{0, 2}**	