

# Aufgabe: ZER

## Nullen und Einsen



BOI 2025, Tag 0. Speicherlimit: 256 MB.

2025.04.25

Es ist eine Folge  $a_0, \dots, a_{n-1}$  von Binärzahlen gegeben. Du kennst ihre Elemente nicht, kannst aber nach der Summe zweier beliebiger, verschiedener Elemente fragen. Du sollst nun die Folge rekonstruieren, ohne zu viele solcher Fragen zu stellen.

## Interaktion

Dies ist eine interaktive Aufgabe. Du sollst ein Programm schreiben, das die korrekte Lösung für die Aufgabe findet und mit dem Grader durch Lesen von der Standardeingabe und schreiben auf die Standardausgabe kommuniziert.

Zu Beginn der Interaktion soll dein Programm eine Ganzzahl  $n$  ( $3 \leq n \leq 200\,000$ ) von der Standardeingabe lesen – die Länge der Folge.

Dann soll dein Programm auf folgender Art und Weise Fragen stellen:

- Dein Programm soll eine einzelne Zeile der Form

$i \ j$

auf die Standardausgabe schreiben, wobei  $i$  und  $j$  verschiedene Ganzzahlen ( $0 \leq i \neq j \leq n-1$ ) sind. Diese Zeile beschreibt eine einzelne Frage zu den Folgengliedern  $a_i$  und  $a_j$ .

- Die Antwort wird gegeben als:

$s$

wobei  $s$  eine Ganzzahl ( $0 \leq s \leq 2$ ) ist, die der Summe  $a_i + a_j$  entspricht.

Sobald du die Folge ermittelt hast, soll dein Programm zwei Zeilen der Form

$n$

$a_0 \ a_1 \ \dots \ a_{n-1}$

auf die Standardausgabe schreiben, wobei  $n$  die Länge der Folge und  $a_0, a_1, \dots, a_{n-1}$  die Folgenglieder sind. **Dann soll die Ausführung deines Programms beendet werden. Weitere Anfragen können in einer Bewertung mit Wrong answer resultieren.**

Denk daran, den Output Buffer nach jeder Anfrage oder Antwort mithilfe von `cout.flush()` (oder `fflush(stdout)`, wenn du `printf` verwendest) in C++ bzw. `sys.stdout.flush()` in Python zu flushen. Andernfalls wird dein Programm möglicherweise mit Time Limit Exceeded bewertet.

Dein Programm darf keine Dateien oder anderen Ressourcen öffnen. Für die Fehlersuche kannst Du den Standard error-Stream verwenden. Bedenke aber, dass das Schreiben auf diesen Stream einige Zeit benötigt.

## C++, input/output mit Streams

Du solltest den passenden Header (`#include <iostream>`) einbinden. Du solltest jede Zeile des Outputs mit `std::endl` beenden. Sieh dir am besten das folgende Beispiel für den Beginn einer Interaktion an.

```
std::cin >> n;
std::cout << i << ' ' << j << std::endl;
std::cin >> sum;
```

## C++, input/output mit stdio

Du solltest den passenden Header (`#include <stdio.h>`) einbinden. Nach der Ausgabe einer Zeile solltest du `fflush(stdout)` aufrufen. Sieh dir am besten das folgende Beispiel für den Beginn einer Interaktion an.

```
scanf("%d", &n);
printf("%d %d\n", i, j);
fflush(stdout);
scanf("%d", &sum);
```

## Python

Jeder Aufruf der print-Funktion sollte `flush = True` als Argument haben. Sieh dir am besten das folgende Beispiel für den Beginn einer Interaktion an.

```
n = int(input())
print(f"{i} {j}", flush = True)
sum = int(input())
```

## Beispielinteraktion

Aktion	Parameter	Ausgabe	Beschreibung
lies $n$	–	5	$n = 5$
summiere	0 1	1	$a_0 + a_1 = 1$
summiere	1 2	1	$a_1 + a_2 = 1$
summiere	3 4	2	$a_3 + a_4 = 2$ , also $a_3 = a_4 = 1$
summiere	0 3	2	$a_0 + a_3 = 2$ , also $a_0 = 1$ , und außerdem $a_1 = 0$ und $a_2 = 1$
antworte	5 1 0 1 1 1		Korrekte Antwort, $m = 4 \leq n = 5$ Fragen, 100% der Punkte.

## Bewertung

Teilaufgabe	Beschränkungen	Punkte
1	$3 \leq n \leq 1000$	50
2	$3 \leq n \leq 200\,000$	50

Der Grader legt die gesamte Folge  $a_0, \dots, a_{n-1}$  **nicht** zu Beginn der Interaktion fest. Stattdessen kann er die Elemente der aktuellen Folge verändern, solange dies den Ergebnissen der vorherigen Fragen nicht widerspricht. In anderen Worten: Der Grader ist adaptiv.

Ist  $m$  die maximale Anzahl von Anfragen, die dein Programm stellt, so erhältst du den folgenden Prozentsatz der Punkte für den jeweiligen Test:

Anfragen	Punktzahl
$m \leq n$	100% der Punkte für den Test
$m = n + 1$	80% der Punkte für den Test
$n + 1 < m \leq n^2 - n$	50% der Punkte für den Test
$m > n^2 - n$	0% der Punkte für den Test ( <b>wrong answer</b> )