D. Gjettelek

Oppgavenavn	Guessing Game
Tidsbegrensning	4 sekund
Minnebegrensning	1 gigabyte

I den gamle byen Lund er det en gate med N hus på rekke, indeksert fra 0 til $N-1$. Emma bor i et av disse husene, og vennene hennes Anna og Bertil vil finne ut hvilket. I stedet for å bare fortelle vennene hennes hvor hun bor, bestemmer Emma seg for å spille et spill med dem. Før spillet starter vet Anna og Bertil bare antallet hus i gaten. På dette tidspunktet kan Anna og Bertil velge et positivt heltall K og bli enig om en strategi. All kommunikasjon etter dette er forbudt.

Spillet i seg selv består av to faser. I den første fasen velger Emma en rekkefølge å besøke husene på slik at hennes er det siste som besøkes. Så leder hun Anna til husene én etter én, uten å fortelle Anna rekkefølgen på forhånd. For hvert hus som ikke er Emma sitt skriver Anna et heltall mellom 1 og K på inngangsdøren. På det siste huset de besøker, som er Emma sitt hus, skriver Emma selv ett heltall mellom 1 og K på døren.

I den andre fasen av spillet går Bertil langs gata fra hus 0 til hus $N-1$ og leser alle tallene skrevet på dørene av Anna og Emma. Han vil nå forsøke å gjette hvilket hus Emma bor i. Han har to forsøk på å gjette rett, og han og Anna vinner spillet om han får det til. Ellers vinner Emma spillet.

Kan du komme opp med en strategi hvor Anna og Bertil er garantert å vinne? Din strategi vil scores på verdien av K (jo lavere, jo bedre).

Implementasjon

Dette er en multi-run oppgave, som betyr at programmet ditt vil bli kjørt flere ganger. Den første gangen det kjøres vil det utføre Annas strategi. Andre gangen vil det utføre Bertils.

Første linje av input vil inneholde to heltall P og N, hvor P er enten 1 eller 2 (første eller andre fase), og N alltid er antall hus. **Sett bort fra i test-inputtet (ikke brukt for scoring) vil N alltid være lik** 100000.

Resterende input avhenger av fasen:

Fase 1

Programmet ditt skal starte med å skrive ut tallet K på en enkelt linje. $(1 \leq K \leq 1\,000\,000)$. Deretter, $N-1$ ganger, skal det lese en linje som inneholder en indeks i $(0 \leq i < N)$ og skrive ut en linje med et tall A_i $(1 \leq A_i \leq K)$ hvor A_i er nummeret Anna skriver på døren til hus i. Hver indeks i, utenom indeksen til Emmas hus, vil opptre her nøyaktig én gang i en rekkefølge bestemt av *graderen*.

Fase 2

Programmet ditt leser en linje med N heltall, $A_0, A_1, ..., A_{N-1}$, hvor A_i er tallet skrevet på hus i.

Deretter skal det skrive ut en linje med to heltall s_1 og s_2 $(0 \leq s_i < N)$, de to indeksene gjettet. s_1 og s_2 har lov til å være like.

Implementasjonsdetaljer

Merk at når programmet kjører i Fase 2 blir programmet omstartet. Dette betyr at du ikke kan lagre informasjon i noen variabler mellom kjøringer.

Etter hver linje skrevet ut, forsikre deg om å flushe standard output. Ellers kan programmet ditt klassifiseres som Time Limit Exceeded. I python flusher `print()` automatisk. I C++ flusher `cout << endl;` også automatisk, sammen med å printe et linjeskift. Anvender du printf, bruk `fflush(stdout)`.

Graderen for dette programmet kan være **adaptiv**. Dette betyr at den kan endre sin oppførsel avhengig av outputtet av ditt program, for å unngå at heuristiske løsninger blir godkjent. Det kan altså kjøre fase 1, se på ditt output, så kjøre fase 1 på nytt og bruke informasjon fra den første kjøringen.

Programmet ditt *må* være deterministisk. Altså må det oppføre seg likt om kjørt to ganger på samme input. Om du ønsker å bruke tilfeldigheter i programmet ditt, forsikre deg om å spesifisere ett *random seed*. I C++ kan dette gjøres ved å sende en hardkodet konstant til `srand`. I Python kan dette gjøres ved å sende en hardkodet konstant til `random.seed`. Spesifikt kan du for eksempel ikke bruke `srand (time(NULL))` i C++. Om graderen oppdager at programmet ditt ikke er deterministisk vil besvarelsen din bli klassifisert som Wrong Answer.

Hvis *summen* av kjøretiden til de (opp til 3) separate kjøringene av programmet ditt overgår tidsbegrensningen, vil besvarelsen din bli klassifisert til Time Limit Exceeded.

Poenggiving

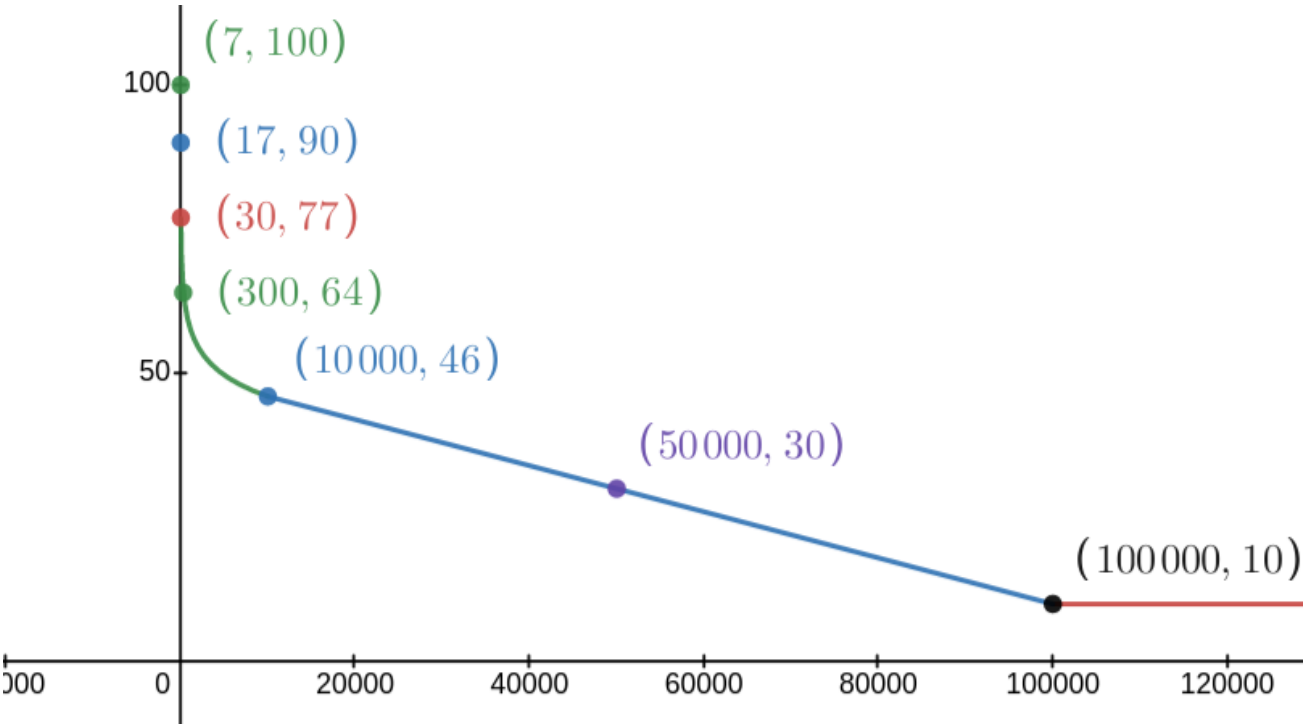
Løsningen din vil testes mot et antall testtilfeller. Hvis løsningen din gjør feil på *hvilken som helst* av disse testtilfellene (for eksempel ved å gi feil svar (Wrong Answer), kræsje (Run-Time Error), overgå

tidsbegrensningen (Time Limit Exceeded), osv.) mottar du 0 poeng og den relevante klassifikasjonen.

Hvis programmet ditt korrekt finner den hemmelige indeksen i *alle* testtilfeller, klassifieres innsendelsen som Accepted, og en poengsum beregnes på følgende måte. La K_{max} være den største verdien K anvendt for noe testtilfelle. Avhengig av K_{max} får du disse poeng:

	Score
$K_{max} > 99\,998$	10 poeng
$10\,000 < K_{max} \leq 99\,998$	$10 + \lfloor 40(1 - K_{max}/10^5) \rfloor$ poeng
$30 < K_{max} \leq 10\,000$	$46 + \lfloor 31(4 - \log_{10}(K_{max}))/(4 - \log_{10}(30)) \rfloor$ poeng
$7 < K_{max} \leq 30$	$107 - K_{max}$ poeng
$K_{max} \leq 7$	100 poeng

Scorefunksjonen er tegnet i figuren under.



Eksempeltesten er ignorert for scoring, og programmet ditt må ikke fungere på denne testen.

Testverktøy

For å muliggjøre testing av løsningen din, gjør vi tilgjengelig et enkelt verktøy du kan laste ned. Se "attachments" på bunnen av oppgavesiden på Kattis. Verktøyet er frivillig å bruke, og du er tillatt å endre det. Merk at det offisielle graderprogrammet på Kattis er et annet program enn dette verktøyet.

Eksempelbruk (med $N = 4$, $s = 2$ hvor s er tallet skrevet på det siste huset):

For Pythonprogrammer, si `solution.py` (normalt kjørt som `pypy3 solution.py`):

```
python3 testing_tool.py pypy3 solution.py <<<"4 2"
```

For C++ programs, kompiler først (for eksempel med `g++ -g -O2 -std=gnu++17 -static solution.cpp -o solution.out`) og kjør så:

```
python3 testing_tool.py ./solution.out <<<"4 2"
```

Testverktøyet vil besøke hus i tilfeldig rekkefølge. For å bruke en spesifikk rekkefølge, endre verktøyet det det står "MODIFY HERE".

Eksempel-interaksjon

Eksempeltesten er ignorert for poenggiving, og løsningen din må ikke fungere på den.

Anta at vi har $N = 4$ og at Emma bor i hus 1. La A være listen med tall skrevet på hus. I utgangspunktet er $A = [0, 0, 0, 0]$, hvor 0 betyr at intet tall er skrevet på det korresponderende huset.

I den første kjøringen av koden din:

$N = 4$ er gitt. Løsningen din svarer med $K = 3$.

A_2 er spurt etter. Løsningen din svarer med 3. A er nå $[0, 0, 3, 0]$.

A_0 er spurt etter. Løsningen din svarer med 1. A er nå $[1, 0, 3, 0]$.

A_3 er spurt etter. Løsningen din svarer med 2. A er nå $[1, 0, 3, 2]$.

Til slutt setter graderen $A_1 = 2$, slik at $A = [1, 2, 3, 2]$ til slutt. Dette markerer slutten på fase en.

I den andre fasen, mottar programmet ditt listen `1 2 3 2`.

Det responderer med `1 3`.

Da et av gjettene er den riktige indeksen til huset (1) vinner Anna og Bertil.

grader output	ditt output
1 4	
	3
2	
	3
0	
	1
3	
	2

grader output	ditt output
2 4	
1 2 3 2	
	1 3