Romfart

Som en del av sikring av en fremtid for mennesker etter jordens uunngåelige undergang, skal N satellitter sendes mot en måne ved navn Fremtid. Fremtid og jorden ligger et ukjent antall lyssekunder unna hverandre. Å reise ett lyssekund koster én enhet med drivstoff.

Fysikere har nylig funnet ut at avstanden mellom jorden og Fremtid er et heltall antall lyssekunder x slik at $A \leq x \leq B$. I tillegg vet de at sannsynligheten for at $x = i$ for i mellom A og B er konstant, altså kan vi si at avstanden fra jorden til Fremtid kan modelleres som et uniformt tilfeldig trukket heltall mellom A og B.

Menneskeheten ønsker å bruke minst mulig drivstoff på å sende N satellitter mot Fremtid. Om Fremtid er x lyssekunder unna jorden vil en satellitt utrustet med minst x enheter drivstoff trygt ankomme Fremtid, mens en satellitt med mindre enn x enheter drivstoff vil forsvinne på vei til Fremtid. Enten ved at satelitten lander eller går tom for drivstoff, vil det sendes ut et signal om dette tilbake til jorden. Det er et ubegrenset antall satellitter på jorden, men drivstoff er en begrenset ressurs. Du har fått i oppdrag å lage en algoritme som bestemmer hvor mye drivstoff som skal sendes på hver av satellittene, én etter én, helt til N av dem har rapportert at de er landet på Fremtid.

Fysikerne på teamet vil teste løsningen din mot flere hypotetiske verdier av A, B og N, før de eventuelt bestemmer om de skal ta i bruk løsningen din i den ekte verden.

Input

Første linje i input består av tallet R, som bestemmer hvor mange runder programmet ditt vil kjøres.

Hver runde starter med en linje input, som består av tallene A, B og N. Input etter dette avhenger av interaksjon.

Interaksjon

For å sende en rakett med k enheter drivstoff, skriv ut `send k`. Avhenging av om satellitten lander eller ikke vil det skrives enten `landet` eller `forsvunnet`.

En runde avsluttes etter at `landet` er skrevet ut fra graderen N ganger.

Merk at stdout blir bufret, det kan derfor være nødvendig å flushe bufferen for hver skriving for å unngå forsinkelser i interaksjonen. I C++ bør du avslutte linjer du printer med `std::endl`. I Python kan du legge til `flush=True` i `print`, eller kalle `sys.stdout.flush()`. I Java flushes stdout automatisk hvis du bruker `System.out.println()`.

Begrensninger

$1 \leq A \leq B \leq 1000$

$1 \leq N \leq 1000$

$1 \leq R \leq 100$

Tidsbegrensning: 2 s

Scoring

I deloppgave 1, 2 og 3 får du poeng etter hvor nær løsningen din er NIO sin løsning. Hvis løsningen din bruker mindre eller like mye drivstoff som NIO sin løsning, så vil du få full uttelling. Deloppgavene består av flere testcaser. Scoren din i en gitt deloppgave er lik summen av scoren for hver testcase, så lenge løsningen din i alle tilfeller gir gyldige svar. Om løsningen din i en test-case i snitt bruker x prosent mer drivstoff en NIO sin løsning, får du $T \cdot 2^{-x}$ poeng i den testcasen, hvor T er maksimalt oppnåelig antall poeng for den spesifike testcasen. Dette betyr for eksempel at om du bruker 1% mer drivstoff enn NIO sin løsning, får du halvparten av full uttelling i testcasen. Husk at hver testcase består av flere runder, så summen av drivstoffbruk for hver test-case består av summen over alle disse rundene.

I deloppgave 4 får du poeng etter hvor nær du er den teoretisk optimale løsningen som innebærer å korrekt tippe avstand mellom jorden og Fremtid. Om du i snitt i en test-case bruker x prosent mer drivstoff enn løsningen som inebærer å korrekt tippe avstanden, får du $T \cdot 2^{-x/10}$ poeng, hvor T er totalt antall poeng for den test-casen. Dette betyr for eksempel at om du bruker 10% mer drivstoff, får du halvparten av full uttelling i testcasen.

Testsettgruppe	**Poeng**	**Ytterligere begrensninger**
Gruppe 1	4	$N = 1$
Gruppe 2	7	$A = 1, B = 2$
Gruppe 3	55	$N \leq 15, B \leq 50$
Gruppe 4	34	$N \geq 30$

Eksempel

Linjer med $>$ foran er output fra programmet ditt og linjer med $<$ foran er `input` som programmet ditt får.

```
< 2
< 12 15 2
> send 12
< forsvunnet
> send 14
< landet
```

```
> send 13
< landet
< 18 18 1
> send 18
< landet
```