Fléttufræði o.fl.

Talning, Líkindi og Leikir

Atli Fannar Franklín

19. mars 2019

Efnisyfirlit

Hvað er fléttufræði?

- Fléttufræði er mjög vítt svið en það sem við munum skoða í dag er að mest *talning*. Þ.e.a.s. mismunandi aðferðafræði til að ákvarða hvað eru margar samsetningar af einhverju mynstri.

Hvað er fléttufræði?

- Fléttufræði er mjög vítt svið en það sem við munum skoða í dag er að mest *talning*. Þ.e.a.s. mismunandi aðferðafræði til að ákvarða hvað eru margar samsetningar af einhverju mynstri.
- Einföld dæmi eru hlutir eins og fjöldi hlutmengja í mengi, fjöldi hlutmengja af gefinni stærð í mengi, fjöldi umraðana, fjöldu umraðana án fastapunkta o.s.frv.

Hvað er fléttufræði?

- Fléttufræði er mjög vítt svið en það sem við munum skoða í dag er að mest *talning*. Þ.e.a.s. mismunandi aðferðafræði til að ákvarða hvað eru margar samsetningar af einhverju mynstri.
- Einföld dæmi eru hlutir eins og fjöldi hlutmengja í mengi, fjöldi hlutmengja af gefinni stærð í mengi, fjöldi umraðana, fjöldu umraðana án fastapunkta o.s.frv.
- Byrjum samt fyrst á grunnatriðunum.

Grunnreglurnar tvær

- Ljóst er að n leiðir eru til þess að velja einn hlut af n valmöguleikum.

Grunnreglurnar tvær

- Ljóst er að n leiðir eru til þess að velja einn hlut af n valmöguleikum.
- Ef við megum velja einn af n valkostum **eða** einn af m valkostum eru $n + m$ valmöguleikar.

Grunnreglurnar tvær

- Ljóst er að n leiðir eru til þess að velja einn hlut af n valmöguleikum.
- Ef við megum velja einn af n valkostum **eða** einn af m valkostum eru $n + m$ valmöguleikar.
- Ef við megum velja einn af n valkostum **og** einn af m valkostum eru nm valmöguleikar.

Grunnreglurnar tvær

- Ljóst er að n leiðir eru til þess að velja einn hlut af n valmöguleikum.
- Ef við megum velja einn af n valkostum **eða** einn af m valkostum eru $n+m$ valmöguleikar.
- Ef við megum velja einn af n valkostum **og** einn af m valkostum eru nm valmöguleikar.
- Þetta leiðir flest af þessu, bara í mismörgum skrefum.

Að labba að π

- Ef við megum velja milli n hluta, nema okkur er bannað að velja m þeirra eru $n - m$ valkostir.

Að labba að π

- Ef við megum velja milli n hluta, nema okkur er bannað að velja m þeirra eru $n - m$ valkostir.
- Það þýðir að ef við höfum n valkosti og m aðra valkosti en það eru k þeirra sameiginlegir þá eru $n + m - k$ valkostir. Þetta má endurrita sem $|A \cup B| = |A| + |B| - |A \cap B|$.

Að labba að π

- Ef við megum velja milli n hluta, nema okkur er bannað að velja m þeirra eru $n - m$ valkostir.
- Það þýðir að ef við höfum n valkosti og m aðra valkosti en það eru k þeirra sameiginlegir þá eru $n + m - k$ valkostir. Þetta má endurrita sem $|A \cup B| = |A| + |B| - |A \cap B|$.
- Við getum haldið þessu áfram, ef við höfum mengi A, B og C þá er

$$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cup B \cup C|$$

PIE

- Með því að framkvæma þrepun má víkka þetta út á sammengi n mengja $A_1, \ldots, A_n$. Þá fæst regla sem kallast 'Principle of inclusion-exclusion', yfirleitt skrifað PIE.

PIE

- Með því að framkvæma þrepun má víkka þetta út á sammengi n mengja $A_1, \ldots, A_n$. Þá fæst regla sem kallast 'Principle of inclusion-exclusion', yfirleitt skrifað PIE.
- Almenna jafnan verður

$$\left|\bigcup_{i=1}^{n} A_i\right| = \sum_{\substack{J \subseteq \{1,\ldots,n\} \\ J \neq \varnothing}} (-1)^{|J|-1} \left|\bigcap_{j \in J} A_j\right|$$

Gagntækar varpanir

- Margt fleira má telja með einföldum reglum. Við höfum þegar séð að fjöldi hlutmengja endanlegs mengis A sé $2^{|A|}$ en skulum sanna það hér með gagntækinni vörpun.

Gagntækar varpanir

- Margt fleira má telja með einföldum reglum. Við höfum þegar séð að fjöldi hlutmengja endanlegs mengis A sé $2^{|A|}$ en skulum sanna það hér með gagntækinni vörpun.
- Oft til að telja hvað er mikið af einhverju er gott að tengja það við eitthvað annað. Tökum hlutmengi $S \subseteq A$. Röðum stökum A einhvern veginn þ.a. $A = \{a_1, \ldots, a_n\}$ þar sem $n = |A|$. Þá búum við til bitastreng með því að láta fyrsta bitann vera 1 ef $a_1 \in S$, annars 0 og svo eins fyrir hina stafina koll af kolli. Auðvelt er að sjá að þetta skilgreini gagntæka vörpun.

Gagntækar varpanir

- Margt fleira má telja með einföldum reglum. Við höfum þegar séð að fjöldi hlutmengja endanlegs mengis A sé $2^{|A|}$ en skulum sanna það hér með gagntækinni vörpun.
- Oft til að telja hvað er mikið af einhverju er gott að tengja það við eitthvað annað. Tökum hlutmengi $S \subseteq A$. Röðum stökum A einhvern veginn þ.a. $A = \{a_1, \ldots, a_n\}$ þar sem $n = |A|$. Þá búum við til bitastreng með því að láta fyrsta bitann vera 1 ef $a_1 \in S$, annars 0 og svo eins fyrir hina stafina koll af kolli. Auðvelt er að sjá að þetta skilgreini gagntæka vörpun.
- En hvað gefur það okkur? Nú það gefur okkur að það sé jafn mikið af báðum hlutum. Nú er auðvelt að telja bitastrengina því við veljum um 2 hluti n sinnum, svo svarið er 2^n.

Gagntækar varpanir

- Margt fleira má telja með einföldum reglum. Við höfum þegar séð að fjöldi hlutmengja endanlegs mengis A sé $2^{|A|}$ en skulum sanna það hér með gagntækinni vörpun.
- Oft til að telja hvað er mikið af einhverju er gott að tengja það við eitthvað annað. Tökum hlutmengi $S \subseteq A$. Röðum stökum A einhvern veginn þ.a. $A = \{a_1, \dots, a_n\}$ þar sem $n = |A|$. Þá búum við til bitastreng með því að láta fyrsta bitann vera 1 ef $a_1 \in S$, annars 0 og svo eins fyrir hina stafina koll af kolli. Auðvelt er að sjá að þetta skilgreini gagntæka vörpun.
- En hvað gefur það okkur? Nú það gefur okkur að það sé jafn mikið af báðum hlutum. Nú er auðvelt að telja bitastrengina því við veljum um 2 hluti n sinnum, svo svarið er 2^n.
- Æfing: Nota gagntækar varpanir (eða bara talningu) til að sýna að fjöldi leiða til að endurraða n ólíkum stökum er $n! := 1 \cdot 2 \cdot \dots \cdot n$.

nCk

- En hvað með fjölda hlutmengja A sem hafa k stök?

nCk

- En hvað með fjölda hlutmengja A sem hafa k stök?
- Þegar við veljum fyrsta stakið höfum við n valkosti. Næst eru bara $n-1$ stök eftir, svo við getum valið annað stakið á $n-1$ veg, og svo fram vegis. Því fáum við $n(n-1)\dots(n-k+1)$ leiðir til að velja stökin.

nCk

- En hvað með fjölda hlutmengja A sem hafa k stök?
- Þegar við veljum fyrsta stakið höfum við n valkosti. Næst eru bara $n-1$ stök eftir, svo við getum valið annað stakið á $n-1$ veg, og svo fram vegis. Því fáum við $n(n-1)\dots(n-k+1)$ leiðir til að velja stökin.
- En þá erum við búin að gefa okkur röðun á stökunum, sem er ekki til staðar í hlutmengi. Því til að bæta upp fyrir það deilum við með $k!$ því við erum búin að telja hvert hlutmengi $k!$ sinnum, einu sinni fyrir hverja endurröðun þess.

nCk

- En hvað með fjölda hlutmengja A sem hafa k stök?
- Þegar við veljum fyrsta stakið höfum við n valkosti. Næst eru bara $n-1$ stök eftir, svo við getum valið annað stakið á $n-1$ veg, og svo fram vegis. Því fáum við $n(n-1)\dots(n-k+1)$ leiðir til að velja stökin.
- En þá erum við búin að gefa okkur röðun á stökunum, sem er ekki til staðar í hlutmengi. Því til að bæta upp fyrir það deilum við með $k!$ því við erum búin að telja hvert hlutmengi $k!$ sinnum, einu sinni fyrir hverja endurröðun þess.
- Við tökum eftir að $n(n-1)\dots(n-k+1) = \frac{n!}{(n-k)!}$ og fáum þá niðurstöðu að svarið sé $\frac{n!}{(n-k)!k!}$. Þetta eru mikilvægar tölur og eru kallaðir tvíliðustuðlar. Við táknum þetta með $\binom{n}{k}$.

nCk

- En hvað með fjölda hlutmengja A sem hafa k stök?
- Þegar við veljum fyrsta stakið höfum við n valkosti. Næst eru bara $n-1$ stök eftir, svo við getum valið annað stakið á $n-1$ veg, og svo fram vegis. Því fáum við $n(n-1)\ldots(n-k+1)$ leiðir til að velja stökin.
- En þá erum við búin að gefa okkur röðun á stökunum, sem er ekki til staðar í hlutmengi. Því til að bæta upp fyrir það deilum við með $k!$ því við erum búin að telja hvert hlutmengi $k!$ sinnum, einu sinni fyrir hverja endurröðun þess.
- Við tökum eftir að $n(n-1)\ldots(n-k+1) = \frac{n!}{(n-k)!}$ og fáum þá niðurstöðu að svarið sé $\frac{n!}{(n-k)!k!}$. Þetta eru mikilvægar tölur og eru kallaðir tvíliðustuðlar. Við táknum þetta með $\binom{n}{k}$.
- Æfing: Notið gagntækar varpanir (ekki algebru) til að sýna að $\binom{n}{k} = \binom{n}{n-k}$ og að $\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$.

nCk reglur

- Hvernig reiknum við þá út? 'Augljósa leiðin' er að reikna margfeldið $n(n-1)\ldots(n-k+1)$ og deila jafn óðum með tölum úr $k!$.

nCk reglur

- Hvernig reiknum við þá út? 'Augljósa leiðin' er að reikna margfeldið $n(n-1)\dots(n-k+1)$ og deila jafn óðum með tölum úr $k!$.
- Þetta er vissulega gilt en langoftast er tvennt sem gerir þetta að slæmri aðferð. Í fyrsta lagi er $\binom{n}{k}$ svo stórt svo hratt að yfirleitt eigum við að skila svarinu modulo p. Í öðru lagi tekur $\mathcal{O}(\backslash)$ tíma að reikna tvíliðustuðul svo yfirleitt eigum við ekki að reikna svo stóra stuðla.

nCk reglur

- Hvernig reiknum við þá út? 'Augljósa leiðin' er að reikna margfeldið $n(n-1)\dots(n-k+1)$ og deila jafn óðum með tölum úr $k!$.
- Þetta er vissulega gilt en langoftast er tvennt sem gerir þetta að slæmri aðferð. Í fyrsta lagi er $\binom{n}{k}$ svo stórt svo hratt að yfirleitt eigum við að skila svarinu modulo p. Í öðru lagi tekur $\mathcal{O}(\backslash)$ tíma að reikna tvíliðustuðul svo yfirleitt eigum við ekki að reikna svo stóra stuðla.
- Því er almennt best (skv. okkar reynslu) að reikna út töflu af gildum $n!$ modulo p í byrjun keyrslu. Svo þegar við viljum reikna $\binom{n}{k}$ reiknum við bara $n!$ modulo p margfaldað við margföldunarandhverfur $k!$ og $(n-k)!$ modulo p. Þá ef N er stærsti tvíliðustuðullinn í dæminu tekur $\mathcal{O}(N)$ tíma að reikna í byrjun og hver stuðull tekur $\mathcal{O}(\log(n))$ tíma (því margföldunarandhverfur modulo p tekur $\log$ tíma að reikna út).

nCk reikningur

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;

const ll maxn = 1e6;
const ll mod = 1e9 + 7;

ll fact[maxn + 1];

void calcfact() {
    fact[0] = 1;
    for(int i = 1; i <= maxn; ++i) fact[i] = (fact[i - 1] * i) % mod;
}

// mod_inv útfærsla úr síðustu viku

ll nck(ll n, ll k) {
    ll res = fact[n];
    res *= mod_inv(fact[k], mod);
    res %= mod;
    res *= mod_inv(fact[n - k], mod);
    res %= mod;
    return res;
}
```

nCk reglur

- Tvíliðustuðlar eru mikilvægir því $\binom{n}{k}$ segir í raun hvað við getum valið k hluti úr hópi n hluta á marga vegu ef innbyrðis röð k hlutanna sem við veljum skiptir ekki máli.

nCk reglur

- Tvíliðustuðlar eru mikilvægir því $\binom{n}{k}$ segir í raun hvað við getum valið k hluti úr hópi n hluta á marga vegu ef innbyrðis röð k hlutanna sem við veljum skiptir ekki máli.
- En það eru fleiri tölur sem koma oft fyrir í talningu. Við höfum áður talað um Fibonacci-tölur og koma þær oft fyrir í talningu.

nCk reglur

- Tvíliðustuðlar eru mikilvægir því $\binom{n}{k}$ segir í raun hvað við getum valið k hluti úr hópi n hluta á marga vegu ef innbyrðis röð k hlutanna sem við veljum skiptir ekki máli.
- En það eru fleiri tölur sem koma oft fyrir í talningu. Við höfum áður talað um Fibonacci-tölur og koma þær oft fyrir í talningu.
- Við munum ekki staldra lengi við í Fibonacci tölunum en sem dæmi getið þið reynt að sannfæra ykkur um að eftirfarandi dæmi skili af sér Fibonacci-tölunum.

nCk reglur

- Tvíliðustuðlar eru mikilvægir því $\binom{n}{k}$ segir í raun hvað við getum valið k hluti úr hópi n hluta á marga vegu ef innbyrðis röð k hlutanna sem við veljum skiptir ekki máli.
- En það eru fleiri tölur sem koma oft fyrir í talningu. Við höfum áður talað um Fibonacci-tölur og koma þær oft fyrir í talningu.
- Við munum ekki staldra lengi við í Fibonacci tölunum en sem dæmi getið þið reynt að sannfæra ykkur um að eftirfarandi dæmi skili af sér Fibonacci-tölunum.
- n manns sitja í fremstu röð í bíósal. Svo fara allir samtímis fram í hlénu. Þegar þeir koma til baka setjast þeir allir annað hvort í sitt eigið sæti eða sæti við hliðina á sínu upphaflega sæti. Hvað eru margar mögulegar niðurraðanir á bíógestunum eftir hlé?

Catalan-tölur

- Catalan-tölurnar koma víða fyrir í talningu. Til dæmis, hvað eru til mörg ólík full tvíundartré á n hnútum? (Full tvíundartré þýðir að allir hnútar hafa tvö börn eða engin). Annað dæmi er, hvað eru hægt að skrifa niður n pör af svigum á marga vegu þ.a. þeir parist rétt saman?

Catalan-tölur

- Catalan-tölurnar koma víða fyrir í talningu. Til dæmis, hvað eru til mörg ólík full tvíundartré á n hnútum? (Full tvíundartré þýðir að allir hnútar hafa tvö börn eða engin). Annað dæmi er, hvað eru hægt að skrifa niður n pör af svigum á marga vegu þ.a. þeir parist rétt saman?
- Ef við táknum svarið fyrir n í þessum spurningum með C_n eiga þau það sameiginlegt að við getum alltaf tekið einn hlut í burtu og svo skipt restinni í tvo hópa og endurkvæmt gert sama strúktúr á þeim, þ.e.

$$C_n = \sum_{i=0}^{n-1} C_i C_{n-i-1}$$

Catalan-tölur

- Catalan-tölurnar koma víða fyrir í talningu. Til dæmis, hvað eru til mörg ólík full tvíundartré á n hnútum? (Full tvíundartré þýðir að allir hnútar hafa tvö börn eða engin). Annað dæmi er, hvað eru hægt að skrifa niður n pör af svigum á marga vegu þ.a. þeir parist rétt saman?
- Ef við táknum svarið fyrir n í þessum spurningum með C_n eiga þau það sameiginlegt að við getum alltaf tekið einn hlut í burtu og svo skipt restinni í tvo hópa og endurkvæmt gert sama strúktúr á þeim, þ.e.

$$C_n = \sum_{i=0}^{n-1} C_i C_{n-i-1}$$

- Ef reiknað er upp úr þessu eru fyrstu tölurnar $1, 1, 2, 5, 14, 42, 132, \ldots$

Catalan-jafna

- Það að finna formúlu fyrir C_n er svolítið meira mál en við höfum tíma fyrir hér. Mæli eindregið með fléttufræði áfanganum (Anders kennir hann eins og er), þar fáið þið að læra margar leiðir til að leiða út formúlu fyrir C_n.

Catalan-jafna

- Það að finna formúlu fyrir C_n er svolítið meira mál en við höfum tíma fyrir hér. Mæli eindregið með fléttufræði áfanganum (Anders kennir hann eins og er), þar fáið þið að læra margar leiðir til að leiða út formúlu fyrir C_n.
- En að því að ég er svo næs ætla ég bara að gefa ykkur að $C_n = \frac{1}{n+1}\binom{2n}{n}$, þetta er langþægilegasta formúlan til að reikna út C_n í tölvu.

Catalan-jafna

- Það að finna formúlu fyrir C_n er svolítið meira mál en við höfum tíma fyrir hér. Mæli eindregið með fléttufræði áfanganum (Anders kennir hann eins og er), þar fáið þið að læra margar leiðir til að leiða út formúlu fyrir C_n.
- En að því að ég er svo næs ætla ég bara að gefa ykkur að $C_n = \frac{1}{n+1}\binom{2n}{n}$, þetta er langþægilegasta formúlan til að reikna út C_n í tölvu.
- En C_n telur margt fleira. Fleiri dæmi eru fjöldi leiða til að skipta n-hyrningi í þríhyrninga með hornalínum, fjöldi umraðana á n stökum sem raða má með hlaða, fjöldi vega frá $(0, 0)$ til (n, n) sem fara heiltöluskref og aldrei niðurfyrir $x = y$ og margt margt fleira.

Umraðanir

- Skoðum aðeins umraðanir betur. Oft er þægilegt að tákna umröðun með **gagntæku** falli $\pi : [n] \to [n]$ þar sem $[n] := \{1, \ldots, n\}$.

Umraðanir

- Skoðum aðeins umraðanir betur. Oft er þægilegt að tákna umröðun með **gagntæku** falli $\pi : [n] \to [n]$ þar sem $[n] := \{1, \ldots, n\}$.
- Það eru $n!$ umraðanir á $[n]$, en það er langt frá því að vera öll sagan. Fastapunktur umraðanar er x þ.a. $\pi(x) = x$. Dæmi um eitt sem má skoða er, hvað eru margar umraðanir á $[n]$ með enga fastapunkta? Svona umraðanir kallast á ensku 'derangement' (stæ.is/os brást mér, fann ekki þýðingu).

Umraðanir

- Skoðum aðeins umraðanir betur. Oft er þægilegt að tákna umröðun með **gagntæku** falli $\pi : [n] \to [n]$ þar sem $[n] := \{1, \ldots, n\}$.
- Það eru $n!$ umraðanir á $[n]$, en það er langt frá því að vera öll sagan. Fastapunktur umraðanar er x þ.a. $\pi(x) = x$. Dæmi um eitt sem má skoða er, hvað eru margar umraðanir á $[n]$ með enga fastapunkta? Svona umraðanir kallast á ensku 'derangement' (stæ.is/os brást mér, fann ekki þýðingu).
- Þá skulum við nota PIE! En við beitum öðru trikki líka sem kallast á ensku 'counting the complement'.

Umraðanir

- Skoðum aðeins umraðanir betur. Oft er þægilegt að tákna umröðun með **gagntæku** falli $\pi : [n] \to [n]$ þar sem $[n] := \{1, \dots, n\}$.
- Það eru $n!$ umraðanir á $[n]$, en það er langt frá því að vera öll sagan. Fastapunktur umraðanar er x þ.a. $\pi(x) = x$. Dæmi um eitt sem má skoða er, hvað eru margar umraðanir á $[n]$ með enga fastapunkta? Svona umraðanir kallast á ensku 'derangement' (stæ.is/os brást mér, fann ekki þýðingu).
- Þá skulum við nota PIE! En við beitum öðru trikki líka sem kallast á ensku 'counting the complement'.
- Við vitum hvað það eru margar umraðanir í heildina, svo við skulum telja hvað það eru margar umraðanir með *einhvern* fastapunkt.

Fastir punktar

- Skoðum nú bara umraðanir á $[n]$. Táknum með A_i allar umraðanir þar sem i er fastapunktur. Þá getum við umraðað restinni eins og við viljum, svo $|A_i| = (n-1)!$. Einnig ef við tökum $\bigcap_{j \in J} A_j$ erum við að festa $|J|$ punkta og getum þá valið restina eins og við viljum, svo $|\bigcap j \in J A_j| = (n - |J|)!$.

Fastir punktar

- Skoðum nú bara umraðanir á $[n]$. Táknum með A_i allar umraðanir þar sem i er fastapunktur. Þá getum við umraðað restinni eins og við viljum, svo $|A_i| = (n-1)!$. Einnig ef við tökum $\bigcap_{j \in J} A_j$ erum við að festa $|J|$ punkta og getum þá valið restina eins og við viljum, svo $|\bigcap j \in J A_j| = (n - |J|)!$.
- Því þegar við summum yfir öll hlutmengi í PIE getum við tekið saman öll hlutmengi af sömu stærð. En við vitum hvað þau eru mörg, það eru $\binom{n}{k}$ af stærð k. Þá gefur PIE okkur beint að fjöldi umraðana með einhvern fastapunkt sé

$$\sum_{i=1}^{n} (-1)^{i-1} \binom{n}{i} (n-i)! = n! \sum_{i=1}^{n} (-1)^{i-1} \frac{1}{i!}$$

Derangement

- En þá er fjöldi derangement-a bara $n!$ að þessu frádregnu. Við táknum þessa stærð með $!n$ og fáum við því að

$$!n = \sum_{i=0}^{n} \frac{(-1)^i}{i!}$$

Derangement

- En þá er fjöldi derangement-a bara $n!$ að þessu frádregnu. Við táknum þessa stærð með $!n$ og fáum við því að

$$!n = \sum_{i=0}^{n} \frac{(-1)^i}{i!}$$

- Skemmtilegt að benda á að nýta má sér veldaröð e^x til að sýna að hlutfall umraðana sem hefur engan fastapunkt stefnir mjög hratt á e^{-1} þegar n stækkar.

Umhverfingar og formerki

- Umraðanir á $[n]$ hafa marga fleiri áhugaverða fléttufræðilega eiginleika en það tvennt sem kemur langoftast fyrir í keppnisforritun eru formerki (sign) umraðana og umhverfingatala (inversion number) þeirra.

Umhverfingar og formerki

- Umraðanir á $[n]$ hafa marga fleiri áhugaverða fléttufræðilega eiginleika en það tvennt sem kemur langoftast fyrir í keppnisforritun eru formerki (sign) umraðana og umhverfingatala (inversion number) þeirra.
- Tökum eftir að ef σ er umröðun getum við skoðað $x, \sigma(x), \sigma(\sigma(x)), \ldots$. Þar sem $[n]$ er endanegt endurtekur þetta sig að lokum og fáum við þá rás í umröðuninni.

Umhverfingar og formerki

- Umraðanir á $[n]$ hafa marga fleiri áhugaverða fléttufræðilega eiginleika en það tvennt sem kemur langoftast fyrir í keppnisforritun eru formerki (sign) umraðana og umhverfingatala (inversion number) þeirra.
- Tökum eftir að ef σ er umröðun getum við skoðað $x, \sigma(x), \sigma(\sigma(x)), \ldots$. Þar sem $[n]$ er endanegt endurtekur þetta sig að lokum og fáum við þá rás í umröðuninni.
- Með því að skoða hvaða rás sérhvert stak tilheyrir má þátta umraðanir í rásir. Þannig fæst rásaframsetning umröðunar, þá er umröðun rituð sem t.d. $(xyz)(ab)(t)$ sem þýðir að $\sigma(x) = y$, $\sigma(y) = z$, $\sigma(z) = x$, $\sigma(a) = b$, $\sigma(b) = a$ og $\sigma(t) = t$. Takið eftir að innihald tveggja sviga skarast ekki.

Umhverfingar og formerki

- Umraðanir á $[n]$ hafa marga fleiri áhugaverða fléttufræðilega eiginleika en það tvennt sem kemur langoftast fyrir í keppnisforritun eru formerki (sign) umraðana og umhverfingatala (inversion number) þeirra.
- Tökum eftir að ef σ er umröðun getum við skoðað $x, \sigma(x), \sigma(\sigma(x)), \ldots$. Þar sem $[n]$ er endanegt endurtekur þetta sig að lokum og fáum við þá rás í umröðuninni.
- Með því að skoða hvaða rás sérhvert stak tilheyrir má þátta umraðanir í rásir. Þannig fæst rásaframsetning umröðunar, þá er umröðun rituð sem t.d. $(xyz)(ab)(t)$ sem þýðir að $\sigma(x) = y$, $\sigma(y) = z$, $\sigma(z) = x$, $\sigma(a) = b$, $\sigma(b) = a$ og $\sigma(t) = t$. Takið eftir að innihald tveggja sviga skarast ekki.
- Ennfremur má rita $(abc \ldots x)$ sem $(ab)(ac) \ldots (ax)$, svo rita má allar umraðanir sem samskeytingu umskiptinga. Sanna má (takið Algebru I ef þið viljið vita meira) að *áferð* (slétt eða odda) fjölda þessarra umhverfinga ákvarðast ótvírætt útfrá gefnu π og köllum við þá formerki π jákvætt ef það er slétt tala og neikvætt ef það er oddatala.

Formerki

- Ennfremur má rita $(abc \dots x)$ sem $(ab)(ac)\dots(ax)$, svo rita má allar umraðanir sem samskeytingu umskiptinga. Sanna má (takið Algebru I ef þið viljið vita meira) að *áferð* (slétt eða odda) fjölda þessarra umhverfinga ákvarðast ótvírætt útfrá gefnu π og köllum við þá formerki π jákvætt ef það er slétt tala og neikvætt ef það er oddatala.

Formerki

- Ennfremur má rita $(abc \ldots x)$ sem $(ab)(ac) \ldots (ax)$, svo rita má allar umraðanir sem samskeytingu umskiptinga. Sanna má (takið Algebru I ef þið viljið vita meira) að *áferð* (slétt eða odda) fjölda þessarra umhverfinga ákvarðast ótvírætt útfrá gefnu π og köllum við þá formerki π jákvætt ef það er slétt tala og neikvætt ef það er oddatala.
- Við táknum þetta með $\operatorname{sgn}(\pi)$ og er það 1 ef það er jákvætt og -1 ef það er neikvætt. Þá fæst beint að $\operatorname{sgn}(\pi_1 \circ \pi_2) = \operatorname{sgn}(\pi_1) \operatorname{sgn}(\pi_2)$ og $\operatorname{sgn}(\mathrm{id}) = 1$. Þetta gefur þá einnig $\operatorname{sgn}(\pi^{-1}) = \operatorname{sgn}(\pi)$.

Formerki

- Ennfremur má rita $(abc\dots x)$ sem $(ab)(ac)\dots(ax)$, svo rita má allar umraðanir sem samskeytingu umskiptinga. Sanna má (takið Algebru I ef þið viljið vita meira) að *áferð* (slétt eða odda) fjölda þessarra umhverfinga ákvarðast ótvírætt útfrá gefnu π og köllum við þá formerki π jákvætt ef það er slétt tala og neikvætt ef það er oddatala.
- Við táknum þetta með $\operatorname{sgn}(\pi)$ og er það 1 ef það er jákvætt og -1 ef það er neikvætt. Þá fæst beint að $\operatorname{sgn}(\pi_1 \circ \pi_2) = \operatorname{sgn}(\pi_1)\operatorname{sgn}(\pi_2)$ og $\operatorname{sgn}(\mathrm{id}) = 1$. Þetta gefur þá einnig $\operatorname{sgn}(\pi^{-1}) = \operatorname{sgn}(\pi)$.
- Þetta kemur stundum fyrir í keppnisforritun sem óbreyta (invariant) í einhverri aðgerð, meir um það síðar.

Umhverfingar

- Tökum $i, j \in [n]$. Raðaða parið (i, j) kallast umhverfing í π af $i < j$ en $\pi(i) > \pi(j)$.

Umhverfingar

- Tökum $i, j \in [n]$. Raðaða parið (i, j) kallast umhverfing í π af $i < j$ en $\pi(i) > \pi(j)$.
- Umhverfingatala π er þá fjöldi svona para. En þó það sé skilgreining tölunnar telur hún margt fleira. Það sem hen telur oftast í keppnisforritun (til viðbótar við fjöldi umhverfinga) hvað þarf að skipta á aðlægum stökum oft til þess að fá umröðunina.

Umhverfingar

- Tökum $i, j \in [n]$. Raðaða parið (i, j) kallast umhverfing í π af $i < j$ en $\pi(i) > \pi(j)$.
- Umhverfingatala π er þá fjöldi svona para. En þó það sé skilgreining tölunnar telur hún margt fleira. Það sem hen telur oftast í keppnisforritun (til viðbótar við fjöldi umhverfinga) hvað þarf að skipta á aðlægum stökum oft til þess að fá umröðunina.
- Sáum áðan að rita má π sem samskeytingu umhverfinga (a, b). En rita má

$$(a\ b) = (a\ a+1)(a+1\ a+2)\ldots(b-1\ b)(b-2\ b-1)\ldots(a\ a+1)$$

Umhverfingar

- Tökum $i, j \in [n]$. Raðaða parið (i, j) kallast umhverfing í π af $i < j$ en $\pi(i) > \pi(j)$.
- Umhverfingatala π er þá fjöldi svona para. En þó það sé skilgreining tölunnar telur hún margt fleira. Það sem hen telur oftast í keppnisforritun (til viðbótar við fjöldi umhverfinga) hvað þarf að skipta á aðlægum stökum oft til þess að fá umröðunina.
- Sáum áðan að rita má π sem samskeytingu umhverfinga (a, b). En rita má

$$(a\ b) = (a\ a+1)(a+1\ a+2)\dots(b-1\ b)(b-2\ b-1)\dots(a\ a+1)$$

- Því má rita π sem samskeytingu umhverfinga á forminu $(x\ x+1)$. Fá má (takið Algebru I til að fá að sjá meira um það) að fjöldi slíkra umhverfinga sem π þáttast í er umhverfingatalan. Þetta kemur annars lagið fyrir í keppnisforritun, yfirleitt sem eitthvað afbrigði af 'hvað þarf að svissa á aðlægum stökum oft til að fá gefna umröðun'.

Að reikna umhverfingatölu

- En hvernig má reikna umhverfingatölu?

Að reikna umhverfingatölu

- En hvernig má reikna umhverfingatölu?
- Það eru tvær algengar leiðir til þess. Annars vegar má keyra mergesort á röðina og telja hvað margar umhverfingar eru 'leiðréttar' í röðuninni jafnóðum. Hins vegar má líka setja stökin inn í Fenwick tré og telja þá jafn óðum hvað eru margar umhverfingar.

Að reikna umhverfingatölu

- En hvernig má reikna umhverfingatölu?
- Það eru tvær algengar leiðir til þess. Annars vegar má keyra mergesort á röðina og telja hvað margar umhverfingar eru 'leiðréttar' í röðuninni jafnóðum. Hins vegar má líka setja stökin inn í Fenwick tré og telja þá jafn óðum hvað eru margar umhverfingar.
- Sjáum hér bæði.

Invnum - Fenwick útgáfa

```
ll invnum(vi &v) {
    ll res = 0;
    vi cnt(v.size() + 1), w(v);
    sort(w.begin(), w.end());
    map<int,int> rnk;
    for(int i = 0; i < v.size(); ++i)
        rnk[w[i]] = i + 1;
    for(int i = v.size() - 1; i >= 0; --i) {
        int j = rnk[v[i]] - 1;
        while(j) res += cnt[j], j -= j & -j;
        j = rnk[v[i]];
        while(j <= v.size()) cnt[j]++, j += j & -j;
    }
    return res;
}
```

Invnum - Mergesort útgáfa

```
ll merge(vi& v, vi& l, vi& r) {
    ll i = 0, j = 0, cnt = 0;
    while(i < l.size() || j < r.size()) {
        if(i == l.size()) v[i + j] = r[j], ++j;
        else if(j == r.size()) v[i + j] = l[i], ++i;
        else if(l[i] <= r[j]) v[i + j] = l[i], ++i;
        else v[i + j] = r[j], cnt += l.size() - i, ++j;
    }
    return cnt;
}

ll invnum(vi &v) {
    if(v.size() < 2) return 0;
    int m = v.size() / 2;
    vi l(m), r(v.size() - m);
    copy(v.begin(), v.begin() + m, l.begin());
    copy(v.begin() + m, v.end(), r.begin());
    return invnum(l) + invnum(r) + merge(v, l, r);
}
```

Óbreytur

- Í mörgum fléttufræðikeppnisforritunardæmum (og leikjafræðidæmum sérstaklega) er oft 'trikkið' að finna óbreytu.

Óbreytur

- Í mörgum fléttufræðikeppnisforritunardæmum (og leikjafræðidæmum sérstaklega) er oft 'trikkið' að finna óbreytu.
- Óbreyta er bara eitthvað sem breytist ekki þegar einhverri aðgerð eða einhverjum aðgerðum er beitt á stöðu.

Óbreytur

- Í mörgum fléttufræðikeppnisforritunardæmum (og leikjafræðidæmum sérstaklega) er oft 'trikkið' að finna óbreytu.
- Óbreyta er bara eitthvað sem breytist ekki þegar einhverri aðgerð eða einhverjum aðgerðum er beitt á stöðu.
- Þetta gæti verið einhver summa sem helst föst meðan heildinni er breytt, einhvert margfeldi sem er fast meðan einhver leikur er spilaður.

Óbreytur

- Í mörgum fléttufræðikeppnisforritunardæmum (og leikjafræðidæmum sérstaklega) er oft 'trikkið' að finna óbreytu.
- Óbreyta er bara eitthvað sem breytist ekki þegar einhverri aðgerð eða einhverjum aðgerðum er beitt á stöðu.
- Þetta gæti verið einhver summa sem helst föst meðan heildinni er breytt, einhvert margfeldi sem er fast meðan einhver leikur er spilaður.
- Náskylt þessu eru einhalla stærðir. Mjög oft ef maður á að svara hvort leikur taki einhvern tímann enda eða eittvað ferli hætti einhvern tímann er 'trikkið' að finna heiltölustærð sem má ekki vera neikvæð en verður að minnka í hvert skipti (eða eitthvað í þann anda).

Óbreytur

- Í mörgum fléttufræðikeppnisforritunardæmum (og leikjafræðidæmum sérstaklega) er oft 'trikkið' að finna óbreytu.
- Óbreyta er bara eitthvað sem breytist ekki þegar einhverri aðgerð eða einhverjum aðgerðum er beitt á stöðu.
- Þetta gæti verið einhver summa sem helst föst meðan heildinni er breytt, einhvert margfeldi sem er fast meðan einhver leikur er spilaður.
- Náskylt þessu eru einhalla stærðir. Mjög oft ef maður á að svara hvort leikur taki einhvern tímann enda eða eittvað ferli hætti einhvern tímann er 'trikkið' að finna heiltölustærð sem má ekki vera neikvæð en verður að minnka í hvert skipti (eða eitthvað í þann anda).
- Sjáum dæmi um þetta.

Ónefnt kattis dæmi

- Stytt lýsing: Gefin röðun á $[n]$, er hægt að búa til aðra gefna röðun á $[n]$ með því að beita bara eftirfarandi aðgerð ($3 \leq n \leq 100000$). Aðgerðin er að taka 3 tölur og færa hana lengst til hægra megin af þessum þremur vinstra megin við hinar tvær og hliðra þá báðar hinar um eitt sæti til hægri.

Ónefnt kattis dæmi

- Stytt lýsing: Gefin röðun á $[n]$, er hægt að búa til aðra gefna röðun á $[n]$ með því að beita bara eftirfarandi aðgerð ($3 \leq n \leq 100000$). Aðgerðin er að taka 3 tölur og færa hana lengst til hægra megin af þessum þremur vinstra megin við hinar tvær og hliðra þá báðar hinar um eitt sæti til hægri.
- Er hægt að finna einhverjar óbreytur í þessu?

Ónefnt kattis dæmi

- Stytt lýsing: Gefin röðun á $[n]$, er hægt að búa til aðra gefna röðun á $[n]$ með því að beita bara eftirfarandi aðgerð ($3 \leq n \leq 100000$). Aðgerðin er að taka 3 tölur og færa hana lengst til hægra megin af þessum þremur vinstra megin við hinar tvær og hliðra þá báðar hinar um eitt sæti til hægri.
- Er hægt að finna einhverjar óbreytur í þessu?
- Formerki umraðanarinnar! Við beitum umröðun á forminu $(a\ b\ c) = (a\ b)(a\ c)$ sem er jákvæð, svo við breytum aldrei formerki umraðanarinnar.

Ónefnt kattis dæmi

- Stytt lýsing: Gefin röðun á $[n]$, er hægt að búa til aðra gefna röðun á $[n]$ með því að beita bara eftirfarandi aðgerð ($3 \leq n \leq 100000$). Aðgerðin er að taka 3 tölur og færa hana lengst til hægra megin af þessum þremur vinstra megin við hinar tvær og hliðra þá báðar hinar um eitt sæti til hægri.
- Er hægt að finna einhverjar óbreytur í þessu?
- Formerki umraðanarinnar! Við beitum umröðun á forminu $(a\ b\ c) = (a\ b)(a\ c)$ sem er jákvæð, svo við breytum aldrei formerki umraðanarinnar.
- Við vitum að við getum ritað umröðunina sem samskeytingu para $(a\ a+1)$. Við viljum því sýna að ef við höfum sléttan fjölda svona umraðana getum við ritað þær sem samskeytingu umraðana á forminu $(a\ a+1\ a+2)$.

Ónefnt kattis dæmi frh.

- Tökum því tvö pör $(a\ a+1)$ og $(b\ b+1)$. Ef $a = b$ er samskeyting þeirra hlutleysan, sem er samskeyting engra þrennda. Ef $a+1 = b$ er samskeytingin $(a\ a+1\ a+2)$. Því þurfum við bara að skoða tilvikið $a+1 < b$ (pörin víxlast svo við megum g.r.f. $a < b$).

Ónefnt kattis dæmi frh.

- Tökum því tvö pör $(a\ a+1)$ og $(b\ b+1)$. Ef $a = b$ er samskeyting þeirra hlutleysan, sem er samskeyting engra þrennda. Ef $a+1 = b$ er samskeytingin $(a\ a+1\ a+2)$. Því þurfum við bara að skoða tilvikið $a+1 < b$ (pörin víxlast svo við megum g.r.f. $a < b$).
- Tökum fyrst eftir $(a+2\ a+1\ a) = (a\ a+1\ a+2)(a\ a+1\ a+2)$. Svo ritum við bara

$$(a\ a+1)(b\ b+1) = (a+1\ a\ a-1)(a\ a-1\ a-2)\dots(b-1\ b\ b+1)$$

Ónefnt kattis dæmi frh.

- Tökum því tvö pör $(a\ a+1)$ og $(b\ b+1)$. Ef $a = b$ er samskeyting þeirra hlutleysan, sem er samskeyting engra þrennda. Ef $a+1 = b$ er samskeytingin $(a\ a+1\ a+2)$. Því þurfum við bara að skoða tilvikið $a+1 < b$ (pörin víxlast svo við megum g.r.f. $a < b$).
- Tökum fyrst eftir $(a+2\ a+1\ a) = (a\ a+1\ a+2)(a\ a+1\ a+2)$. Svo ritum við bara

$$(a\ a+1)(b\ b+1) = (a+1\ a\ a-1)(a\ a-1\ a-2)\dots(b-1\ b\ b+1)$$

- Svo ef umröðunin er slétt má fá hana með þessum hætti! Því er svarið bara hvort umraðaninar tvær hafi sama formerki!

Lausnin

```cpp
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
typedef vector<int> vi;

// invnum útfærsla

int main() {
    ll n, x;
    cin >> n;
    vi p1inv(n), p2(n), p(n);
    for(int i = 0; i < n; ++i) {
        cin >> x;
        p1inv[--x] = i;
    }
    for(int i = 0; i < n; ++i) {
        cin >> p2[i];
        p2[i]--;
    }
    for(int i = 0; i < n; ++i) {
        p[i] = p2[p1inv[i]];
    }
    cout << (invnum(p) % 2 == 0 ? "Possible\n" : "Impossible\n");
}
```

Talningadæmi (good luck að finna það á kattis)

- Tökum líka sýnidæmi um talningu. Fáum $1 \leq n \leq 1000$ og $1 \leq c \leq 10000$ og eigum að finna fjölda umraðana á n stökum með umhverfingatölu c. Skila á svarinu modulo $10^9 + 7$.

Talningadæmi (good luck að finna það á kattis)

- Tökum líka sýnidæmi um talningu. Fáum $1 \leq n \leq 1000$ og $1 \leq c \leq 10000$ og eigum að finna fjölda umraðana á n stökum með umhverfingatölu c. Skila á svarinu modulo $10^9 + 7$.
- Hvar byrjum við eiginlega?

Talningadæmi (good luck að finna það á kattis)

- Tökum líka sýnidæmi um talningu. Fáum $1 \leq n \leq 1000$ og $1 \leq c \leq 10000$ og eigum að finna fjölda umraðana á n stökum með umhverfingatölu c. Skila á svarinu modulo $10^9 + 7$.
- Hvar byrjum við eiginlega?
- Getum við smækkað vandamálið fyrir n og c niður í smærri vandamál?

Talningadæmi (good luck að finna það á kattis)

- Tökum líka sýnidæmi um talningu. Fáum $1 \leq n \leq 1000$ og $1 \leq c \leq 10000$ og eigum að finna fjölda umraðana á n stökum með umhverfingatölu c. Skila á svarinu modulo $10^9 + 7$.
- Hvar byrjum við eiginlega?
- Getum við smækkað vandamálið fyrir n og c niður í smærri vandamál?
- Það getum við!

Talningadæmi (good luck að finna það á kattis)

- Táknum svarið fyrir n, c með $f(n, c)$. Skrifum niður umröðunina okkar sem lista $\pi(1), \ldots, \pi(n)$. Við getum búið til umröðun á $n + 1$ staki með því að bæta við $n + 1$ einhversstaðar í listann. Ef við bætum því við í sæti k (og hliðrum öllu til hægri til að mynda pláss) þá myndast $n - k + 1$ umhverfingar.

Talningadæmi (good luck að finna það á kattis)

- Táknum svarið fyrir n, c með $f(n, c)$. Skrifum niður umröðunina okkar sem lista $\pi(1), \ldots, \pi(n)$. Við getum búið til umröðun á $n+1$ staki með því að bæta við $n+1$ einhversstaðar í listann. Ef við bætum því við í sæti k (og hliðrum öllu til hægri til að mynda pláss) þá myndast $n-k+1$ umhverfingar.
- Eins getum við gert þetta afturábak. Ef við höfum umröðun á n stökum getum við fjarlægt n úr listanum til að fá umröðun á $n-1$ staki. Skiptum nú í tilvik. S.s. reynum að skoða fjölda umraðana á n stökum með k umhverfingar þar sem n er í sæti t.

Talningadæmi (good luck að finna það á kattis)

- Táknum svarið fyrir n, c með $f(n, c)$. Skrifum niður umröðunina okkar sem lista $\pi(1), \ldots, \pi(n)$. Við getum búið til umröðun á $n+1$ staki með því að bæta við $n+1$ einhversstaðar í listann. Ef við bætum því við í sæti k (og hliðrum öllu til hægri til að mynda pláss) þá myndast $n-k+1$ umhverfingar.
- Eins getum við gert þetta afturábak. Ef við höfum umröðun á n stökum getum við fjarlægt n úr listanum til að fá umröðun á $n-1$ staki. Skiptum nú í tilvik. S.s. reynum að skoða fjölda umraðana á n stökum með k umhverfingar þar sem n er í sæti t.
- Þá við að fjarlægja n fjarlægjum við $t-1$ umhverfingu. En þá höfum við umröðun á $n-1$ staki eftir sem hefur $c-t+1$ umhverfingar. Því fyrir þetta fasta t er svarið $f(n-1, c-t+1)$.

Talningadæmi (good luck að finna það á kattis)

- Táknum svarið fyrir n, c með $f(n, c)$. Skrifum niður umröðunina okkar sem lista $\pi(1), \ldots, \pi(n)$. Við getum búið til umröðun á $n+1$ staki með því að bæta við $n+1$ einhversstaðar í listann. Ef við bætum því við í sæti k (og hliðrum öllu til hægri til að mynda pláss) þá myndast $n-k+1$ umhverfingar.
- Eins getum við gert þetta afturábak. Ef við höfum umröðun á n stökum getum við fjarlægt n úr listanum til að fá umröðun á $n-1$ staki. Skiptum nú í tilvik. S.s. reynum að skoða fjölda umraðana á n stökum með k umhverfingar þar sem n er í sæti t.
- Þá við að fjarlægja n fjarlægjum við $t-1$ umhverfingu. En þá höfum við umröðun á $n-1$ staki eftir sem hefur $c-t+1$ umhverfingar. Því fyrir þetta fasta t er svarið $f(n-1, c-t+1)$.
- Því með því að summa yfir $t = 1, \ldots, n$ fæst að

$$f(n,k) = \sum_{i=0}^{\min(c,n-1)} f(n-1, c-i)$$

Kvik bestun

- Því gerum við nú bara töflu yfir öll $f(n, c)$-in og reiknum þau út. En það er $\mathcal{O}(\backslash^{\in}\rfloor)$ því það eru nc stök í töflunni og hvert tekur n tíma. Þetta er rétt svo ekki nógu gott.

Kvik bestun

- Því gerum við nú bara töflu yfir öll $f(n,c)$-in og reiknum þau út. En það er $\mathcal{O}(\backslash^{\in}\rfloor)$ því það eru nc stök í töflunni og hvert tekur n tíma. Þetta er rétt svo ekki nógu gott.
- Því til að spara okkur tíma gerum við aðra töflu $p(n,c)$ sem er skilgreind með

$$p(n,c) = \sum_{i=0}^{c} f(n,i)$$

Kvik bestun

- Því gerum við nú bara töflu yfir öll $f(n, c)$-in og reiknum þau út. En það er $\mathcal{O}(\backslash^{\in}\rfloor)$ því það eru nc stök í töflunni og hvert tekur n tíma. Þetta er rétt svo ekki nógu gott.
- Því til að spara okkur tíma gerum við aðra töflu $p(n, c)$ sem er skilgreind með

$$p(n, c) = \sum_{i=0}^{c} f(n, i)$$

- Þá getum við notað þá töflu til þess að reikna út hvert $f(n, c)$ í föstum tíma og eins fyrir $p(n, c)$. Þá verður tímaflækjan $\mathcal{O}(nc)$ sem er nóg!

Lausnin

```
#include <bits/stdc++.h>
using namespace std;
const int mod = 1e9 + 7;

int n, c;
int dp[1005][10005];
int pr[1005][10005];

int main() {
    cin >> n >> c;
    if(c == 0) {
        cout << "1\n";
        return 0;
    }
    for(int i = 0; i <= c; ++i) dp[0][i] = 0;
    for(int i = 0; i < n; ++i) dp[i][0] = 1;
    for(int i = 1; i < n; ++i) {
        pr[i - 1][0] = dp[i - 1][0];
        for(int j = 1; j <= c; ++j) {
            pr[i - 1][j] = (pr[i - 1][j - 1] + dp[i - 1][j]) % mod;
        }
        for(int j = 1; j <= c; ++j) {
            dp[i][j] = pr[i - 1][j];
            if(j >= i + 1) {
                dp[i][j] -= pr[i - 1][j - i - 1];
                dp[i][j] = (dp[i][j] % mod + mod) % mod;
            }
        }
    }
    cout << dp[n - 1][c] << '\n';
}
```

Efnisyfirlit

Líkindafræði

- Við ætlum að tala örsnöggt um líkindafræði því dæmi eiga það til að fjalla um hana.

Líkindafræði

- Við ætlum að tala örsnöggt um líkindafræði því dæmi eiga það til að fjalla um hana.
- Langoftast byggja dæmin á mjög einfaldri líkindafræði og þyngd dæmisins kemur úr því að reikna upp úr líkindafræðijöfnunum með hröðum hætti (yfirleitt þá með kvikri bestun).

Líkindafræði

- Við ætlum að tala örsnöggt um líkindafræði því dæmi eiga það til að fjalla um hana.
- Langoftast byggja dæmin á mjög einfaldri líkindafræði og þyngd dæmisins kemur úr því að reikna upp úr líkindafræðijöfnunum með hröðum hætti (yfirleitt þá með kvikri bestun).
- Stundum koma þyngri dæmi um slembiferli, en við höfum ekki tíma til að skoða það hér. Áhugasamir geta talað við Berg um slembiferli eða bara tekið slembiferlaáfangann hér í HÍ.

Grunnatriði

- Við ætlum að tala örsnöggt um líkindafræði því dæmi eiga það til að fjalla um hana.

Grunnatriði

- Við ætlum að tala örsnöggt um líkindafræði því dæmi eiga það til að fjalla um hana.
- Langoftast byggja dæmin á mjög einfaldri líkindafræði og þyngd dæmisins kemur úr því að reikna upp úr líkindafræðijöfnunum með hröðum hætti (yfirleitt þá með kvikri bestun).

Grunnatriði

- Við ætlum að tala örsnöggt um líkindafræði því dæmi eiga það til að fjalla um hana.
- Langoftast byggja dæmin á mjög einfaldri líkindafræði og þyngd dæmisins kemur úr því að reikna upp úr líkindafræðijöfnunum með hröðum hætti (yfirleitt þá með kvikri bestun).
- Stundum koma þyngri dæmi um slembiferli, en við höfum ekki tíma til að skoða það hér. Áhugasamir geta talað við Berg um slembiferli eða bara tekið slembiferlaáfangann hér í HÍ.

Grunnatriði

- Við ætlum að tala örsnöggt um líkindafræði því dæmi eiga það til að fjalla um hana.
- Langoftast byggja dæmin á mjög einfaldri líkindafræði og þyngd dæmisins kemur úr því að reikna upp úr líkindafræðijöfnunum með hröðum hætti (yfirleitt þá með kvikri bestun).
- Stundum koma þyngri dæmi um slembiferli, en við höfum ekki tíma til að skoða það hér. Áhugasamir geta talað við Berg um slembiferli eða bara tekið slembiferlaáfangann hér í HÍ.
- Við ætlum ekki heldur að skoða formlegar skilgreiningar á líkindum, líkindamáli, dreififöllum o.s.frv. Ef þið viljið kynnast slíkum hlutum getiði tekið lík og töl og grundvöll líkindafræðinnar.

Strjál grunnatriði

- Í keppnisforritun rekumst við á tvennskonar líkindi, samfelld og strjál. Skulum skoða strjál fyrst.

Strjál grunnatriði

- Í keppnisforritun rekumst við á tvennskonar líkindi, samfelld og strjál. Skulum skoða strjál fyrst.
- Þá eru einhverjir endanlega margir hlutir (eða teljanlega margir) sem geta gerst $x_1, \ldots, x_n$ og fyrir hvert þeirra eru líkindi $0 \leq P(x_i) \leq 1$ á því að það gerist þ.a. $\sum P(x_i) = 1$ því það eru 100% líkur á því að *eitthvað* gerist. **Atburður** X er þá mengi svona x_i-a og eru líkurnar á því að hann gerist líkurnar á því að eitthvert x_i-anna gerist.

Strjál grunnatriði

- Í keppnisforritun rekumst við á tvennskonar líkindi, samfelld og strjál. Skulum skoða strjál fyrst.
- Þá eru einhverjir endanlega margir hlutir (eða teljanlega margir) sem geta gerst $x_1, \ldots, x_n$ og fyrir hvert þeirra eru líkindi $0 \leq P(x_i) \leq 1$ á því að það gerist þ.a. $\sum P(x_i) = 1$ því það eru 100% líkur á því að *eitthvað* gerist. **Atburður** X er þá mengi svona x_i-a og eru líkurnar á því að hann gerist líkurnar á því að eitthvert x_i-anna gerist.
- Líkurnar á því að x_1 og x_2 gerist er þá $P(x_1)P(x_2)$. Einnig eru líkurnar á því að x_1 eða x_2 gerist $P(x_1) + P(x_2) - P(x_1)P(x_2)$. Líkurnar á að x_1 gerist *ekki* er þá $1 - P(x_i)$. Þessar reglur gilda ekki almennt um atburði því ef X og Y eru atburðir og $X \cap Y \neq \varnothing$ er $P(X \cap Y)$ ekki endilega jafnt $P(X)P(Y)$.

Strjál grunnatriði

- Í keppnisforritun rekumst við á tvennskonar líkindi, samfelld og strjál. Skulum skoða strjál fyrst.
- Þá eru einhverjir endanlega margir hlutir (eða teljanlega margir) sem geta gerst $x_1, \ldots, x_n$ og fyrir hvert þeirra eru líkindi $0 \leq P(x_i) \leq 1$ á því að það gerist þ.a. $\sum P(x_i) = 1$ því það eru 100% líkur á því að *eitthvað* gerist. **Atburður** X er þá mengi svona x_i-a og eru líkurnar á því að hann gerist líkurnar á því að eitthvert x_i-anna gerist.
- Líkurnar á því að x_1 og x_2 gerist er þá $P(x_1)P(x_2)$. Einnig eru líkurnar á því að x_1 eða x_2 gerist $P(x_1) + P(x_2) - P(x_1)P(x_2)$. Líkurnar á að x_1 gerist *ekki* er þá $1 - P(x_i)$. Þessar reglur gilda ekki almennt um atburði því ef X og Y eru atburðir og $X \cap Y \neq \varnothing$ er $P(X \cap Y)$ ekki endilega jafnt $P(X)P(Y)$.
- Ef tákna má x_i-in með tölum, þ.e. $x_i \in \mathbb{R}$ þá getum við talað um væntigildi. Þ.e. ef við veljum eitthvað x_i með þessum gefnu líkindum, hvað fáum við að meðaltali út stóra tölu. Það er gefið með $E[X] = \sum x_i P(x_i)$.

Væntigildi og samfelldni

- Það sem er mikilvægasti eiginleiki væntigildis (eini sem er nauðsynlegur í keppnisforritun) er að hann er línulegur. Þ.e. ef X og Y og a rauntala þá er $E[X+Y] = E[X] + E[Y]$ og $E[aX] = aE[X]$. Þetta gildir almennt, ekki bara í strjála tilvikinu.

Væntigildi og samfelldni

- Það sem er mikilvægasti eiginleiki væntigildis (eini sem er nauðsynlegur í keppnisforritun) er að hann er línulegur. Þ.e. ef X og Y og a rauntala þá er $E[X+Y] = E[X] + E[Y]$ og $E[aX] = aE[X]$. Þetta gildir almennt, ekki bara í strjála tilvikinu.
- En samfelld líkindi eru líka til, þá er mengi allra útkoma óteljanlega óendanlegt. Þegar þetta er gert kemur heilmikil fræði inn málið, en sem betur fer fyrir okkur eru einu samfelldu líkindin í keppnisforritun (sem við höfum séð) jafndreifðu líkindin $U([a,b])$. Þetta er þegar útkoman getur verið hvaða rauntala í $[a,b]$ sem er og þær eru allar jafn líklegar.

Væntigildi og samfelldni

- Það sem er mikilvægasti eiginleiki væntigildis (eini sem er nauðsynlegur í keppnisforritun) er að hann er línulegur. Þ.e. ef X og Y og a rauntala þá er $E[X+Y] = E[X] + E[Y]$ og $E[aX] = aE[X]$. Þetta gildir almennt, ekki bara í strjála tilvikinu.
- En samfelld líkindi eru líka til, þá er mengi allra útkoma óteljanlega óendanlegt. Þegar þetta er gert kemur heilmikil fræði inn málið, en sem betur fer fyrir okkur eru einu samfelldu líkindin í keppnisforritun (sem við höfum séð) jafndreifðu líkindin $U([a,b])$. Þetta er þegar útkoman getur verið hvaða rauntala í $[a,b]$ sem er og þær eru allar jafn líklegar.
- Svo þegar $X \sim U([a,b])$ er $E[X] = \frac{b-a}{2}$ því að meðaltali lendum við í meðaltalinu. Formlega þyrfti að sýna þetta með heildun ($\int_a^b dx$).

Væntigildi og samfelldni

- Það sem er mikilvægasti eiginleiki væntigildis (eini sem er nauðsynlegur í keppnisforritun) er að hann er línulegur. Þ.e. ef X og Y og a rauntala þá er $E[X+Y] = E[X] + E[Y]$ og $E[aX] = aE[X]$. Þetta gildir almennt, ekki bara í strjála tilvikinu.
- En samfelld líkindi eru líka til, þá er mengi allra útkoma óteljanlega óendanlegt. Þegar þetta er gert kemur heilmikil fræði inn málið, en sem betur fer fyrir okkur eru einu samfelldu líkindin í keppnisforritun (sem við höfum séð) jafndreifðu líkindin $U([a,b])$. Þetta er þegar útkoman getur verið hvaða rauntala í $[a,b]$ sem er og þær eru allar jafn líklegar.
- Svo þegar $X \sim U([a,b])$ er $E[X] = \frac{b-a}{2}$ því að meðaltali lendum við í meðaltalinu. Formlega þyrfti að sýna þetta með heildun ($\int_a^b dx$).
- En satt best að segja er hægt að taka flest keppnisforritunarlíkindadæmi (nema örfá þyngri dæmi) bara á 'tilfinningunni' eða 'innsæinu'. Sjáum sýnidæmi.

Ónefnt kattis sýnidæmi

- Dæmið (stytt útgáfa) er að við sendum skilaboð einvhern tímann milli t_1 og t_2 (t_1, t_2 með). Ef við sendum skilaboðin á tíma t sendum við svo líka skilaboð á tíma $t + m_1, t + m_2, \ldots, t + m_n$. Svo fáum við tímabil $[b_1, e_1], \ldots, [b_n, e_n]$ þar sem ekki má senda skilaboð. Hverjar eru líkurnar á því að við fylgjum þeirri reglu? Höfum $1 \leq n, k \leq 10000$, $nk \leq 10^7$, $0 \leq t_1, t_2, m_i, b_i, e_i \leq 10^{16}$.

Ónefnt kattis sýnidæmi

- Dæmið (stytt útgáfa) er að við sendum skilaboð einvhern tímann milli t_1 og t_2 (t_1, t_2 með). Ef við sendum skilaboðin á tíma t sendum við svo líka skilaboð á tíma $t + m_1, t + m_2, \ldots, t + m_n$. Svo fáum við tímabil $[b_1, e_1], \ldots, [b_n, e_n]$ þar sem ekki má senda skilaboð. Hverjar eru líkurnar á því að við fylgjum þeirri reglu? Höfum $1 \leq n, k \leq 10000$, $nk \leq 10^7$, $0 \leq t_1, t_2, m_i, b_i, e_i \leq 10^{16}$.
- Við sjáum að t er 'gott' ef $t + m_i \notin [b_j, e_j]$ fyrir öll i, j. Sér í lagi er þá t 'slæmt' þ.þ.a.a. til sé i, j þ.a. $t \in [b_j - m_i, e_j, m_i]$.

Ónefnt kattis sýnidæmi

- Dæmið (stytt útgáfa) er að við sendum skilaboð einvhern tímann milli t_1 og t_2 (t_1, t_2 með). Ef við sendum skilaboðin á tíma t sendum við svo líka skilaboð á tíma $t + m_1, t + m_2, \ldots, t + m_n$. Svo fáum við tímabil $[b_1, e_1], \ldots, [b_n, e_n]$ þar sem ekki má senda skilaboð. Hverjar eru líkurnar á því að við fylgjum þeirri reglu? Höfum $1 \leq n, k \leq 10000$, $nk \leq 10^7$, $0 \leq t_1, t_2, m_i, b_i, e_i \leq 10^{16}$.
- Við sjáum að t er 'gott' ef $t + m_i \notin [b_j, e_j]$ fyrir öll i, j. Sér í lagi er þá t 'slæmt' þ.þ.a.a. til sé i, j þ.a. $t \in [b_j - m_i, e_j, m_i]$.
- Reynum því að ákvarða samtals lengd allra bila á forminu $[b_j - m_i, e_j - m_i]$ og drögum það frá $t_2 - t_1$ og deilum svo með $t_2 - t_1$.

Ónefnt kattis sýnidæmi frh.

- Við búum því til lista l af öllum slæmum bilum. Fyrir hvert i, j tökum við $[b_j - m_i, e_j - m_i]$, ef það er sundurlægt $[t_1, t_2]$ hendum við því bara. Annars setjum við $[b_j - m_i, e_j - m_i] \cap [t_1, t_2]$ í listann l.

Ónefnt kattis sýnidæmi frh.

- Við búum því til lista l af öllum slæmum bilum. Fyrir hvert i, j tökum við $[b_j - m_i, e_j - m_i]$, ef það er sundurlægt $[t_1, t_2]$ hendum við því bara. Annars setjum við $[b_j - m_i, e_j - m_i] \cap [t_1, t_2]$ í listann l.
- Við getum þá raðað l eftir upphafspunkta bilsins (og endapunkt ef það eru jafntefli). Svo tökum við fremsta bilið í listanum og setjum það sem breytu c fyrir 'current' og setjum $s = 0$. Svo löbbum við bara í gegnum listann í röð. Fyrir hvert bil kemur tvennt til greina.

Ónefnt kattis sýnidæmi frh.

- Við búum því til lista l af öllum slæmum bilum. Fyrir hvert i, j tökum við $[b_j - m_i, e_j - m_i]$, ef það er sundurlægt $[t_1, t_2]$ hendum við því bara. Annars setjum við $[b_j - m_i, e_j - m_i] \cap [t_1, t_2]$ í listann l.
- Við getum þá raðað l eftir upphafspunkta bilsins (og endapunkt ef það eru jafntefli). Svo tökum við fremsta bilið í listanum og setjum það sem breytu c fyrir 'current' og setjum $s = 0$. Svo löbbum við bara í gegnum listann í röð. Fyrir hvert bil kemur tvennt til greina.
- Ef það er sundurlægt frá c þá bætum við lengd þess við s og setjum nýja bilið sem c. Ef það er ekki sundurlægt frá c sameinum við það við c og geymum niðurstöðuna í c og höldum áfram. Í lokin bætum við svo lengd c við s og er þá s samtals lengdin sem við vildum.

Ónefnt kattis sýnidæmi frh.

- Við búum því til lista l af öllum slæmum bilum. Fyrir hvert i, j tökum við $[b_j - m_i, e_j - m_i]$, ef það er sundurlægt $[t_1, t_2]$ hendum við því bara. Annars setjum við $[b_j - m_i, e_j - m_i] \cap [t_1, t_2]$ í listann l.
- Við getum þá raðað l eftir upphafspunkta bilsins (og endapunkt ef það eru jafntefli). Svo tökum við fremsta bilið í listanum og setjum það sem breytu c fyrir 'current' og setjum $s = 0$. Svo löbbum við bara í gegnum listann í röð. Fyrir hvert bil kemur tvennt til greina.
- Ef það er sundurlægt frá c þá bætum við lengd þess við s og setjum nýja bilið sem c. Ef það er ekki sundurlægt frá c sameinum við það við c og geymum niðurstöðuna í c og höldum áfram. Í lokin bætum við svo lengd c við s og er þá s samtals lengdin sem við vildum.
- Þá er bara svarið $(t_2 - t_1 - s)/(t_2 - t_1)$ og prentum við það út.

Efnisyfirlit

Leikjafræðigrunnur

- Leikjafræði er fólgin í því að við séum með leikmenn að spila leik sem spila ávallt fullkomlega (taka alltaf bestu ákvörðunina frá þeirra bæjardyrum séð) og viljum skoða hvort einhver leikmaður eigi sér vinningsstrategíu fyrir gefnar upphafsstöður.

Leikjafræðigrunnur

- Leikjafræði er fólgin í því að við séum með leikmenn að spila leik sem spila ávallt fullkomlega (taka alltaf bestu ákvörðunina frá þeirra bæjardyrum séð) og viljum skoða hvort einhver leikmaður eigi sér vinningsstrategíu fyrir gefnar upphafsstöður.
- Leikirnir eru þá skilgreindir útfrá einhverjum stöðum, t.d. hvaða spil eru eftir á hendi hjá leikmönnum, hvað eru margir steinar eftir í hrúgu, hvernig leikborð lítur út (eins og fyrir myllu) etc. etc.

Leikjafræðigrunnur

- Leikjafræði er fólgin í því að við séum með leikmenn að spila leik sem spila ávallt fullkomlega (taka alltaf bestu ákvörðunina frá þeirra bæjardyrum séð) og viljum skoða hvort einhver leikmaður eigi sér vinningsstrategíu fyrir gefnar upphafsstöður.
- Leikirnir eru þá skilgreindir útfrá einhverjum stöðum, t.d. hvaða spil eru eftir á hendi hjá leikmönnum, hvað eru margir steinar eftir í hrúgu, hvernig leikborð lítur út (eins og fyrir myllu) etc. etc.
- Þá fyrir hverja stöðu eru einhverjar löglegar hreyfingar. Einnig eru einhverjar stöður tapstöður, fyrir myllu væri það t.d. þegar andstæðingurinn er kominn með 3 í röð.

Leikjafræðigrunnur

- Leikjafræði er fólgin í því að við séum með leikmenn að spila leik sem spila ávallt fullkomlega (taka alltaf bestu ákvörðunina frá þeirra bæjardyrum séð) og viljum skoða hvort einhver leikmaður eigi sér vinningsstrategíu fyrir gefnar upphafsstöður.
- Leikirnir eru þá skilgreindir útfrá einhverjum stöðum, t.d. hvaða spil eru eftir á hendi hjá leikmönnum, hvað eru margir steinar eftir í hrúgu, hvernig leikborð lítur út (eins og fyrir myllu) etc. etc.
- Þá fyrir hverja stöðu eru einhverjar löglegar hreyfingar. Einnig eru einhverjar stöður tapstöður, fyrir myllu væri það t.d. þegar andstæðingurinn er kominn með 3 í röð.
- Fyrir leiki þar sem báðir andstæðingar vita allt (s.s. engin falin spil á hendi og slíkt) þá getum við rakið okkur afturábak til að sjá hvort staða sé vinningsstaða eða ekki.

Rekja sig afturábak

- Við byrjum á að skrá allar lokastöður (s.s. þar sem leikurinn er búinn, annað hvort vegna jafnteflis eða einn vann hinn).

Rekja sig afturábak

- Við byrjum á að skrá allar lokastöður (s.s. þar sem leikurinn er búinn, annað hvort vegna jafnteflis eða einn vann hinn).
- Svo skoðum við stöður endurkvæmt. Fyrir allar stöður sem ég kemst í vel ég það besta fyrir mig. Ef einhver þeirra er tapstaða fyrir andstæðinginn vel ég hana og er þá staðan mín vinningsstaða. Ef engin þeirra er tapstaða reyni ég að velja jafnteflisstöðu og er þá staðan mín jafnteflisstaða. Ef allar stöður eru vinningsstöður fyrir andstæðinginn þá verð ég bara að sætta mig við það að mín staða sé tapstaða.

Rekja sig afturábak

- Við byrjum á að skrá allar lokastöður (s.s. þar sem leikurinn er búinn, annað hvort vegna jafnteflis eða einn vann hinn).
- Svo skoðum við stöður endurkvæmt. Fyrir allar stöður sem ég kemst í vel ég það besta fyrir mig. Ef einhver þeirra er tapstaða fyrir andstæðinginn vel ég hana og er þá staðan mín vinningsstaða. Ef engin þeirra er tapstaða reyni ég að velja jafnteflisstöðu og er þá staðan mín jafnteflisstaða. Ef allar stöður eru vinningsstöður fyrir andstæðinginn þá verð ég bara að sætta mig við það að mín staða sé tapstaða.
- Skoðum einfalt dæmi um þetta (ekki forritunardæmi bara leikjadæmi).

Einfaldur leikur

- Skoðum leik þar sem við byrjum með n steina í hrúgu. Svo skiptast tveir leikmenn A og B á að taka 1 eða 2 steina úr hrúgunni. Sá sem tekur síðasta steininn vinnur. Fyrir hvaða n vinnur A og fyrir hvaða n vinnur B? (Það eru ljóslega engin jafntefli).

Einfaldur leikur

- Skoðum leik þar sem við byrjum með n steina í hrúgu. Svo skiptast tveir leikmenn A og B á að taka 1 eða 2 steina úr hrúgunni. Sá sem tekur síðasta steininn vinnur. Fyrir hvaða n vinnur A og fyrir hvaða n vinnur B? (Það eru ljóslega engin jafntefli).
- Ef þú átt að gera og engir steinar eru eftir taparðu, svo 0 er tapstaða.

Einfaldur leikur

- Skoðum leik þar sem við byrjum með n steina í hrúgu. Svo skiptast tveir leikmenn A og B á að taka 1 eða 2 steina úr hrúgunni. Sá sem tekur síðasta steininn vinnur. Fyrir hvaða n vinnur A og fyrir hvaða n vinnur B? (Það eru ljóslega engin jafntefli).
- Ef þú átt að gera og engir steinar eru eftir taparðu, svo 0 er tapstaða.
- Ef einn eða tveir steinar eru eftir geturðu sett andstæðinginn í tapstöðuna 0 svo 1 og 2 eru vinningsstöður.

Einfaldur leikur

- Skoðum leik þar sem við byrjum með n steina í hrúgu. Svo skiptast tveir leikmenn A og B á að taka 1 eða 2 steina úr hrúgunni. Sá sem tekur síðasta steininn vinnur. Fyrir hvaða n vinnur A og fyrir hvaða n vinnur B? (Það eru ljóslega engin jafntefli).
- Ef þú átt að gera og engir steinar eru eftir taparðu, svo 0 er tapstaða.
- Ef einn eða tveir steinar eru eftir geturðu sett andstæðinginn í tapstöðuna 0 svo 1 og 2 eru vinningsstöður.
- Ef það eru þrír steinar á borðinu þá sama hvort þú tekur einn eða tvo seturðu andstæðinginn í vinningsstöðu, svo 3 tapstaða.

Einfaldur leikur frh.

- Svona má halda áfram með þrepun til að sjá að n sé tapstaða þ.þ.a.a. $n = 0 \ (mod\ 3)$.

Einfaldur leikur frh.

- Svona má halda áfram með þrepun til að sjá að n sé tapstaða þ.þ.a.a. $n = 0 \ (mod\ 3)$.
- Þar sem A byrjar þá fáum við að A tapar þ.þ.a.a. upphaflegur fjöldi steina sé tapstaða.

Einfaldur leikur frh.

- Svona má halda áfram með þrepun til að sjá að n sé tapstaða þ.þ.a.a. $n = 0\ (mod\ 3)$.
- Þar sem A byrjar þá fáum við að A tapar þ.þ.a.a. upphaflegur fjöldi steina sé tapstaða.
- Því er svarið einfaldlega, fyrir $n = 0\ (mod\ 3)$ vinnur B og fyrir allar aðrar stöður vinnur A.

Aðeins flóknari leikur

- Skoðum annan leik þar sem við höfum n hrúgur, hver með $k_1, \ldots, k_n$ steinum. Leikmaður má þá taka eins marga steina og hann vill, en bara úr einni hrúgu. Sá sem tekur síðasta steininn vinnur. Þessi leikur er mjög frægur í leikjafræði og kallast nim.

Aðeins flóknari leikur

- Skoðum annan leik þar sem við höfum n hrúgur, hver með $k_1, \ldots, k_n$ steinum. Leikmaður má þá taka eins marga steina og hann vill, en bara úr einni hrúgu. Sá sem tekur síðasta steininn vinnur. Þessi leikur er mjög frægur í leikjafræði og kallast nim.
- Það er töluvert erfiðara að sjá út hver vinnur núna, en trikkið er að finna mjög sniðuga óbreytu.

Aðeins flóknari leikur

- Skoðum annan leik þar sem við höfum n hrúgur, hver með $k_1, \ldots, k_n$ steinum. Leikmaður má þá taka eins marga steina og hann vill, en bara úr einni hrúgu. Sá sem tekur síðasta steininn vinnur. Þessi leikur er mjög frægur í leikjafræði og kallast nim.
- Það er töluvert erfiðara að sjá út hver vinnur núna, en trikkið er að finna mjög sniðuga óbreytu.
- Trikkið er að þegar þú er búinn að leika viltu að $k_1 \oplus \cdots \oplus k_n = 0$ og að slíkt hið sama gildi aldrei fyrir andstæðinginn. Af hverju er þetta sniðugt?

Aðeins flóknari leikur

- Skoðum annan leik þar sem við höfum n hrúgur, hver með $k_1, \dots, k_n$ steinum. Leikmaður má þá taka eins marga steina og hann vill, en bara úr einni hrúgu. Sá sem tekur síðasta steininn vinnur. Þessi leikur er mjög frægur í leikjafræði og kallast nim.
- Það er töluvert erfiðara að sjá út hver vinnur núna, en trikkið er að finna mjög sniðuga óbreytu.
- Trikkið er að þegar þú er búinn að leika viltu að $k_1 \oplus \cdots \oplus k_n = 0$ og að slíkt hið sama gildi aldrei fyrir andstæðinginn. Af hverju er þetta sniðugt?
- 0 er tapstaða svo ef $k_1 \oplus \cdots \oplus k_n = 0$ eftir hvern leik þá fæst að að lokum verði öll k_i-in núll og getur þá andstæðingurinn aldrei sett þig í þá tapstöðu, en þú vinnur að lokum.

Xorsum

- En hvernig látum við xor-summuna vera 0 eftir okkar leik? Við verðum þá að gera ráð fyrir að hún sé ekki 0 þegar við byrjum að gera.

Xorsum

- En hvernig látum við xor-summuna vera 0 eftir okkar leik? Við verðum þá að gera ráð fyrir að hún sé ekki 0 þegar við byrjum að gera.
- Látum xorsummuna vera X. Skoðum þá allar stærðirnar $X \oplus k_i$. Vegna þess að $\oplus$ er tengið er þetta jafnt xor-summu allra k_j-a nema k_i. Því er til einhver hrúga þ.a. $X \oplus k_j < k_j$ (því $X \neq 0$). Við tökum þá $k_j - X \oplus k_j > 0$ steina úr hrúgu k_j. Þar sem $a \oplus a = 0$ er þá nýja xor-summan $X \oplus k_j \oplus X \oplus k_j = 0$.

Xorsum

- En hvernig látum við xor-summuna vera 0 eftir okkar leik? Við verðum þá að gera ráð fyrir að hún sé ekki 0 þegar við byrjum að gera.
- Látum xorsummuna vera X. Skoðum þá allar stærðirnar $X \oplus k_i$. Vegna þess að $\oplus$ er tengið er þetta jafnt xor-summu allra k_j-a nema k_i. Því er til einhver hrúga þ.a. $X \oplus k_j < k_j$ (því $X \neq 0$). Við tökum þá $k_j - X \oplus k_j > 0$ steina úr hrúgu k_j. Þar sem $a \oplus a = 0$ er þá nýja xor-summan $X \oplus k_j \oplus X \oplus k_j = 0$.
- En ef xor-summan er núll þá sama hvað við tökum marga steina úr hrúgu k_i þ.a. t verði eftir verður $X \oplus k_i \oplus t \neq 0$ því til þess þyrfti $t = X \oplus k_i$ en það er jafnt k_i svo við þyrftum að taka enga steina.

Xorsum

- En hvernig látum við xor-summuna vera 0 eftir okkar leik? Við verðum þá að gera ráð fyrir að hún sé ekki 0 þegar við byrjum að gera.
- Látum xorsummuna vera X. Skoðum þá allar stærðirnar $X \oplus k_i$. Vegna þess að $\oplus$ er tengið er þetta jafnt xor-summu allra k_j-a nema k_i. Því er til einhver hrúga þ.a. $X \oplus k_j < k_j$ (því $X \neq 0$). Við tökum þá $k_j - X \oplus k_j > 0$ steina úr hrúgu k_j. Þar sem $a \oplus a = 0$ er þá nýja xor-summan $X \oplus k_j \oplus X \oplus k_j = 0$.
- En ef xor-summan er núll þá sama hvað við tökum marga steina úr hrúgu k_i þ.a. t verði eftir verður $X \oplus k_i \oplus t \neq 0$ því til þess þyrfti $t = X \oplus k_i$ en það er jafnt k_i svo við þyrftum að taka enga steina.
- Þar með er staða tapstaða þ.þ.a.a. $k_1 \oplus \cdots \oplus k_n = 0$.

Grundy-Sprague

- Skoðum eina setningu svona í lokin sem við sönnum ekki. Hún heitir Grundy-Sprague setningin.

Grundy-Sprague

- Skoðum eina setningu svona í lokin sem við sönnum ekki. Hún heitir Grundy-Sprague setningin.
- Gefum okkur leik þar sem að engin jafntefli eru möguleg, það eru engar huldar upplýsingar, leikur tekur alltaf endanlega marga tíma, fyrir sömu stöður geta báðir leikmenn alltaf leikið sömu leikina og leik er lokið þegar leikmaður getur ekki leikið (og hann tapar þá).

Grundy-Sprague

- Skoðum eina setningu svona í lokin sem við sönnum ekki. Hún heitir Grundy-Sprague setningin.
- Gefum okkur leik þar sem að engin jafntefli eru möguleg, það eru engar huldar upplýsingar, leikur tekur alltaf endanlega marga tíma, fyrir sömu stöður geta báðir leikmenn alltaf leikið sömu leikina og leik er lokið þegar leikmaður getur ekki leikið (og hann tapar þá).
- Þá getum við skilgreint fyrir hverja stöðu svokallaða Grundy-tölu.

Grundy-tölur

- Við byrjum á að gefa öllum lokastöðum þar sem leikmaður tapar Grundy-tölu 0 (t.d. þegar hrúgan er tóm í nim, ekki fyrir allar tapstöður samt endilega).

Grundy-tölur

- Við byrjum á að gefa öllum lokastöðum þar sem leikmaður tapar Grundy-tölu 0 (t.d. þegar hrúgan er tóm í nim, ekki fyrir allar tapstöður samt endilega).
- Skilgreinum svo mex af mengi. Við skilgreinum það sem minnstu ekki neikvæðu töluna sem er ekki í menginu. Við látum þá Grundy-tölu stöðu vera mex af Grundy-tölunum sem hægt er að komast í úr þeirri stöðu.

Grundy-tölur

- Við byrjum á að gefa öllum lokastöðum þar sem leikmaður tapar Grundy-tölu 0 (t.d. þegar hrúgan er tóm í nim, ekki fyrir allar tapstöður samt endilega).
- Skilgreinum svo mex af mengi. Við skilgreinum það sem minnstu ekki neikvæðu töluna sem er ekki í menginu. Við látum þá Grundy-tölu stöðu vera mex af Grundy-tölunum sem hægt er að komast í úr þeirri stöðu.
- Þá er ljóst að staða sé tapstaða þ.þ.a.a. Grundy-talan sé núll. En það mikilvæga er ekki það, heldur það sem kemur á næstu glæru.

Grundy-Sprague

- Ímyndum okkur nú að við höfum k leiki í gangi í einu. Þá má leikmaður bara leika í einum leikjanna þegar hann á að gera. Einnig tapar þá leikmaður þegar hann getur ekki leikið í neinum leikjanna.

Grundy-Sprague

- Ímyndum okkur nú að við höfum k leiki í gangi í einu. Þá má leikmaður bara leika í einum leikjanna þegar hann á að gera. Einnig tapar þá leikmaður þegar hann getur ekki leikið í neinum leikjanna.
- Þá segir Grundy-Sprague setningin að staða í þessum margfalda leik sé tapstaða þ.þ.a.a. xor-summa Grundy-talna staðanna í hverjum leik fyrir sig sé 0, svipað og í nim.

Sýnidæmi af kattis

- Lýsing: Stan and Ollie play the game of multiplication by multiplying an integer p by one of the numbers 2 to 9. Stan always starts with $p = 1$, does his multiplication, then Ollie multiplies the number, then Stan and so on. Before a game starts, they draw an integer n and the winner is who first reaches $p \geq n$.

Sýnidæmi af kattis

- Lýsing: Stan and Ollie play the game of multiplication by multiplying an integer p by one of the numbers 2 to 9. Stan always starts with $p = 1$, does his multiplication, then Ollie multiplies the number, then Stan and so on. Before a game starts, they draw an integer n and the winner is who first reaches $p \geq n$.
- Input: Each line of input contains the integer $1 < n < 4294967295$. There are at most 30 lines of input.

Sýnidæmi af kattis

- Lýsing: Stan and Ollie play the game of multiplication by multiplying an integer p by one of the numbers 2 to 9. Stan always starts with $p = 1$, does his multiplication, then Ollie multiplies the number, then Stan and so on. Before a game starts, they draw an integer n and the winner is who first reaches $p \geq n$.
- Input: Each line of input contains the integer $1 < n < 4294967295$. There are at most 30 lines of input.
- Output: Print 'Stan wins.' if Stan wins and 'Ollie wins.' if Ollie wins, without the quotation marks, assuming both of them play optimally.

Sýnidæmi af kattis

- Hverjar eru mögulegu stöðurnar? Þær eru ljóslega tölur frá 1 upp að $9n$ en getum við takmarkað okkur meira?

Sýnidæmi af kattis

- Hverjar eru mögulegu stöðurnar? Þær eru ljóslega tölur frá 1 upp að $9n$ en getum við takmarkað okkur meira?
- Já! Það eru tölur í $[1, 9n]$ sem eru margfeldi af tölunum $2, 3, \ldots, 9$ sem þýðir að einu frumþættirnir þeirra eru $2, 3, 5, 7$. Eftir því sem n vex verður þetta mjög lágt hlutfall af heildinni, nánar til tekið vex fjöldinn eins og $\log(n)^4$.

Sýnidæmi af kattis

- Hverjar eru mögulegu stöðurnar? Þær eru ljóslega tölur frá 1 upp að $9n$ en getum við takmarkað okkur meira?
- Já! Það eru tölur í $[1, 9n]$ sem eru margfeldi af tölunum $2, 3, \ldots, 9$ sem þýðir að einu frumþættirnir þeirra eru $2, 3, 5, 7$. Eftir því sem n vex verður þetta mjög lágt hlutfall af heildinni, nánar til tekið vex fjöldinn eins og $\log(n)^4$.
- Svo eftir að reikna allar stöðurnar rekjum við okkur bara afturábak (fyrir fast n) og sjáum hvað svarið er!

Sýnidæmi lausn

```
// fastpow er bara int pow fall
// reiknum hér út allar mögulegar stöður
set<ll> poss;
ll mx = 4294967295;

void calcposs() {
    ll num;
    poss.insert(0);
    poss.insert(fastpow(2, 32));
    for(ll p2 = 0; p2 < 31; ++p2) {
        for(ll p3 = 0; p3 < 20; ++p3) {
            for(ll p5 = 0; p5 < 13; ++p5) {
                for(ll p7 = 0; p7 < 11; ++p7) {
                    num = fastpow(2, p2) * fastpow(3, p3) * \
                        fastpow(5, p5) * fastpow(7, p7);
                    if(num > mx) break;
                    poss.insert(num);
                } } } } }
```

Sýnidæmi lausn

```
bool calcwin(ll g) {
    win.clear();
    for(auto it = --poss.upper_bound(g); it != poss.begin(); --it) {
        if(*it * 9 >= g) {
            win[*it] = true;
            continue;
        }
        win[*it] = false;
        for(ll i = 2; i < 10; ++i) {
            win[*it] |= !win[*it * i];
        }
    }
    return win[1];
}

int main() {
    ll n;
    calcposs();
    while(cin >> n) {
        cout << (calcwin(n) ? "Stan wins." : "Ollie wins.") << endl;
    }
}
```