Ime varijable "CLAN" možda nije jako zgodno za opći dolik algoritma pretrage – jer previše asocira na uvjet $x_i = a$, tj. da testiramo baš da li je neki član x_i u nizu jednak zadanom a.

Za opću formu algoritma, tu logičku vrijednost (varijablu) je zgodnije nazvati

"nasli" ili "found" (na engl.)

Onda opći algoritam sekvencijalnog pretraživanja ima oblik:

$i = \emptyset$;
NASLI = laž;
sve dok (**not** NASLI) **and** $(i < n)$ ponavljaj
- $i = i + 1$;
- NASLI = $x_i = a$;

Ovaj algoritam provjerava da li je neki član niza (x_i) jednak zadanom a i to se vidi u zadnjoj naredbi

$$\text{NASLI} = \underbrace{x_i = a}_{\text{ono što tražimo (uvjet pretrage)}}.$$

Po potpuno istom principu možemo tražiti i drugačije stvari (tj. s drugačijim uvjetom).

Umjesto: "jednak zadanom broju a" $(x_i = a)$ mogu staviti

"djeljiv s 5" $(x_i \bmod 5 = \emptyset)$

pa algoritam traži da je neki član niza djeljiv s 5, ili, preciznije, daje odgovor na pitanje

postoji li (bar jedan) član niza sa zadanim svojstvom – djeljiv s 5.

<u>Zadatak</u> Što treba napraviti ako tražim odgovor na pitanje:

da li <u>svaki</u> član niza ima zadano svojstvo?

Tj. "okrenem" kvantifikator postoji ($\exists$) u svaki ($\forall$).

— · —

Vratimo se polaznom problemu provjere da li se zadani objekt a nalazi u nizu podataka $x_1, \ldots, x_n$. Sekvencijalno traženje koristi samo usporedbe

jednak / različit.

Ako objekte možemo uspoređivati "pravom" relacijom uređaja, tj. operacijama

$<, \leq$ odn. $>, \geq$

onda problem pretraživanja možemo <u>drastično ubrzati</u>, tako da niz podataka prvo SORTIRAMO,

uzlazno: $x_1 \leq x_2 \leq \ldots \leq x_n$ ($\nearrow$)

silazno: $x_1 \geq x_2 \geq \ldots \geq x_n$ ($\searrow$).

Za početak, idemo napraviti taj "brzi" algoritam pretrage koji se zove BINARNO PRETRAŽIVANJE, a onda ćemo napraviti nekoliko osnovnih algoritama za sortiranje podataka (nizova, ali i nekih drugačijih struktura – sličnog tipa).

- <u>Brzo</u> traženje (pa i binarno pretraživanje) u <u>sortiranom</u> nizu podataka vrlo nalikči na traženje telefonskog broja zadane osobe u telefonskom imeniku.

- Telefonski imenik je, srećom, sortiran (uzlazno) po onom podatku kojeg najčešće znamo, a to je prezime i ime osobe (pa eventualno i adresu)

(Zamislite da je sortiran po telefonskom broju! Tako su vaše liste obično sortirane po JMBAG-u, a ne po imenu!)

- Ako mene tražite, onda ćete imenik otvoriti negdje na 2/3 od početka (ili čak 3/4) i pogledati prezimena na stranici – recimo prvo (na vrhu lijeve stranice).

Ako je prezime <u>ispred</u> mog ($\leq$), onda me treba tražiti IZA – u stražnjem dijelu imenika. Obratno, ako je prezime iza mog ($\geq$), onda me treba tražiti ISPRED – u prednjem dijelu.

U oba slučaja znamo da me u jednom od dijelova NE treba tražiti (tj. ako me ima, onda sam u onom drugom dijelu.

- Dakle – na <u>nizu brojeva</u> to izgleda ovako:

$$x_1 \leq x_2 \leq \ldots\ldots \leq \underbrace{x_i}_{\text{odabrani objekt (indeks)}} \leq \ldots\ldots \leq x_{n-1} \leq x_n$$

Zadani objekt a uspoređujem s odabranim x_i (malo kasnije – kako biram x_i).
Tu usporedbu mogu napraviti na nekoliko načina – biram relacijski operator za usporedbu x_i i a.

1. "Pedantno" uspoređujem:
   prvo: $a < x_i$, pa: $a > x_i$, i onda: $a = x_i$
   (redoslijed ova 3 "pitanja" nije bitan)

   Što ću dalje – naravno – ovisi o odgovorima na ova pitanja.

Idemo redom:

- ako je $a < x_i$ onda zaključujem:
  - a sigurno NIJE u komadu niza $x_i \leq \ldots \leq x_n$ (straga)
    (i taj dio niza više NE gledam)
  - a može biti samo u komadu

    $x_1 \leq \ldots \leq x_{i-1}$ (sprijeda)

  pa nastavljam traženje u tom komadu.

  Što sam dobio? SKRATIO sam komad niza u kojem tražim. Koliko – ovisi o i, n.

- u protivnom, znam $a \geq x_i$, pa pitam "$a > x_i$".
  ako je $a > x_i$, opet:
  - a sigurno ~~nije~~ NIJE u komadu niza $x_1 \leq \ldots \leq x_i$ (sprijeda)
    (njega više NE gledam!)
  - a može biti samo u komadu

    $x_{i+1} \leq \ldots \leq x_n$ (straga)

  i nastavljam traženje u tom komadu.

  Uoči – opet sam sigurno skratio niz!

- na kraju, ako su oba prethodna pitanja dala negativan odgovor, tj.
  nije $a < x_i$, pa je $\underline{a \geq x_i}$
  a onda nije $a > x_i$, pa mora biti $a = x_i$,
  Što tad? Eureka, našli smo ga!
  (Mogu prestati tražiti).

- Idemo prvo pogledati kako se bira x_i, odnosno i.
  Odgovor: "na pola" komada niza u kojem tražimo, tj. u našem slučaju – kad tražimo u $x_1 \leq \ldots \leq x_n$, onda je

  $$i = \left\lfloor \frac{n+1}{2} \right\rfloor \text{ ili } \left\lceil \frac{n+1}{2} \right\rceil$$

  "indeks polovišta ~~koji~~ od lijevog ruba 1 do desnog ruba n".

Zašto "na pola"?

U danom trenutku, ne znam gdje je a, i očekujem da je podjednako vjerojatno da je

a lijevo od x_i i a desno od x
$(a < x_i)$ $(a > x_i)$

- U prvom slučaju $(a < x_i)$, preostaje pretraga u nizu $x_1 \leq \ldots \leq x_{i-1}$ – duljine

$$i-1,$$

a u drugom slučaju $(a > x_i)$, moramo tražiti u komadu $x_{i+1} \leq \ldots \leq x_n$, duljine

$$n-i.$$

- Ako su ta dva slučaja "podjednako" vjerojatna, isplati se i izabrati tako da u oba slučaja dobijem jednake (ili bar podjednake) duljine nizova koje još treba pretražiti. Dakle, smisleno je i uzeti tako da je

$$i-1 \approx n-i$$
$$=$$

odakle odmah izlazi da je i polovište intervala indeksa $[1, n]$,

$$i \approx \frac{n+1}{2}.$$
$$=$$

Na kraju, i (indeks!) mora biti cijeli broj, pa uzmemo donje ili gornje cijelo

$$i = \left\lfloor \frac{n+1}{2} \right\rfloor \quad \text{ili} \quad i = \left\lceil \frac{n+1}{2} \right\rceil$$

standardni izbor, jer cjelobrojno dijeljenje s 2 daje baš ovaj rezultat!

Dakle, uspoređujem s elementom "na polovištu" komada niza u kojem tražim.

Osim toga, ako je $a \neq x_i$, onda, nakon toga moram tražiti u komadu duljine ispod polovine prethodnog komada:

$$\text{za } i = \left\lfloor \frac{n+1}{2} \right\rfloor \text{ je } \quad i-1 \leq \left\lfloor \frac{n}{2} \right\rfloor \quad (\text{kad je } = ?)$$

$$\text{i} \quad n-i \leq \left\lfloor \frac{n}{2} \right\rfloor \quad (\text{kad je } = ?).$$

Isto vrijedi i za izbor $i = \left\lceil \frac{n+1}{2} \right\rceil$.

- Zaključak – "raspolovio" sam duljinu niza u kojem još moram tražiti.
- Zato se ova pretraga katkad zove i pretraga raspolavljanjem (BISEKCIJOM), a standardni naziv je BINARNO PRETRAŽIVANJE.

(eliminiram ili fiksiram 1 bit u binarnom prikazu indeksa onog člana niza koji je, eventualno, jednak a)

- Dosad smo opisali samo prvi korak pretrage

$$x_1, \dots, x_n \longmapsto \begin{cases} x_1, \dots, x_{i-1} & (\text{za } a < x_i) \\ x_{i+1}, \dots, x_n & (\text{za } a > x_i) \\ x_i = a & \end{cases}$$

I sad ponavljam pretragu na preostalom komadu niza. Uočite njegove ~~...~~ indekse – početak i kraj. (jedan od njih se promijenio, drugi ostaje isti!)

- Za zgodan zapis petlje, isplati se uvesti posebne oznake za:

lijevi rub – indeks prvog elementa (najmanjeg!)
desni rub – indeks zadnjeg –"– (najvećeg)

u komadu niza kojeg još treba pretražiti.

Dogovor: ℓ = lijevi rub, d = desni rub
tj. tražimo a u komadu niza

$$x_\ell \leq \ldots \leq x_d$$

(interval indeksa je $[\ell, d]$).

- Na početku je, naravno,

$$\ell = 1,\ d = n$$

(cijeli niz još treba pretražiti).

- Fale nam još samo uvjeti u petlji za pretragu.
  Prvi je očit – traži dok nismo našli ("pohlepa")
  tj. sve dok (<u>not</u> nasli).
  A onaj drugi – "i još ima gdje tražiti"?
  Zar nije i to očito?
  To znači da preostali niz NIJE PRAZAN (tj. $[\ell, d] \neq \emptyset$)
  odnosno:

$$\underline{\underline{\ell \leq d}}.$$

prazan skup

- Algoritam binarnog pretraživanja glasi:

```
NASLI = laž;
l = 1; d = n;
sve dok (not nasli) and (l ≤ d) ponavljaj
  i = ⌊(l+d)/2⌋;   (div)
  ako je a < x_i onda
     d = i-1;                  [l, i-1]
  inače ako je a > x_i onda
     l = i+1;                  [i+1, d]
  inače
     nasli = istina;
```

Složenost?

Opet brojimo broj prolaza kroz petlju, odnosno broj usporedbi a s nekim članom x_i (s tim da svaka usporedba može imati 2 pitanja, a ne samo jedno!)

Gledam najgori mogući slučaj – a to je da a ne nađem.

Zbog nasli = laz, tada zadnji prolaz kroz petlju ide na nizu duljine najviše 2 (ne može 3), a barem 1.

Sad iskoristimo da se duljina komada kojeg još treba pretražiti smanjuje barem na pola (možda čak i malo više!)

nakon 1. prolaza : $\leq \frac{n}{2}$ ← duljina preostalog niza

nakon 2. prolaza : $\leq \frac{n}{4}$

$\vdots$

nakon k. prolaza : $\leq \frac{n}{2^k}$

I što sad? (Hoću: broj prolaza $k \leq$ nešto!)
Kad sam sigurno napravio zadnji prolaz?

Kad NAKON njega dobijem

$$\frac{n}{2^k} < 1 .$$

(za = 1, još moram u sljedeći prolaz!)

Onda sigurno stajemo. Dakle

$$n < 2^k$$

ili

$$k > \log_2 n .$$

Hm!? Znak je na obratnu stranu!
Polako. Stajem prvi puta kad se to dogodi (daljnjih prolaza nema!). Dakle, trebam najmanji takav k.

A to znači da je u prolazu $k-1$ vrijedilo

$$\frac{n}{2^{k-1}} \geq 1. \quad \left(\begin{array}{l}\text{inače bismo stali}\\ \text{u tom prolazu!}\end{array}\right)$$

$$n \geq 2^{k-1}$$

$$k-1 \leq \log_2 n$$

$$k \leq \log_2 n + 1$$

ili

$$\boxed{k \leq 1 + \lfloor \log_2 n \rfloor}.$$

Gruba provjera: za $n=1$ – treba provjeriti 1 član, pa nam treba 1 prolaz. Dobijem

$$k \leq 1 + \lfloor \underbrace{\log_2 1}_{0} \rfloor = 1 \quad \text{OK}.$$

— · —

Ovaj algoritam smo izveli na osnovu pitanja:

prvo: $a < x_i$, pa $a > x_i$ i onda $a = x_i$ (znamo)

Može nam se dogoditi da 2 puta pitamo za svaki x_i.

Druga alternativa je "tvrda odluka" da pitam samo jednom, recimo ovako:

pitam: $a < x_i$ $\rightarrow$ ako je, ostajem tražiti u

$$x_l \leq \ldots \leq x_{i-1}, \text{ tj. } d = i-1$$

a, ako nije – ne pitam dalje, nego znam da je $a \geq x_i$, pa ostajem tražiti u

$$\underbrace{x_i}_{\substack{\text{ostaje za}\\ \text{pretragu}}} \leq \ldots \leq x_d, \text{ tj. } l = i$$

(NE više $l = i-1$)

Kako sad izlazim van iz petlje i kako postavim vrijednost odgovora (nasli)?

Zbog oblika pitanja, x_i ostaje u "gornjem komadu" (desnom)
(više ga NE eliminiramo!).

Zbroj duljina komada je = početna duljina
(ne smanjuje se za 1).

Zato "režem" sve dok je duljina bar 2, tj. sve dok je $\ell < d$. Iza toga — izlazim s $\ell \geq d$.
I sad oprez — u kojem komadu (duljine 1) još treba gledati?

Odgovor — u <u>gornjem</u> (desnom) — dakle, pitam $a = x_\ell$ ili $a = x_d$?
Pravo pitanje: $a = x_\ell$ ← razlog: $\ell = i$ je uvijek OK, a $d = i-1$ može proći van — $\ell = 1, d = 2, i = 1$, $a < x_1$ pa $d = 0$!
, i to je konačni odgovor.

- Pripadni algoritam:

NASLI = laž;
$\ell = 1$; $d = n$;
sve dok je $\ell < d$ ponavljaj
    $i = \lfloor (\ell + d)/2 \rfloor$; (div)
    ako je $a < x_i$ onda
        $d = i - 1$;
    inače
        $\ell = i$;

NASLI = $(a = x_\ell)$;

- <u>Zadatak</u>: Sastavite analogne varijante algoritma za pitanja: $a \leq x_i$, $a \geq x_i$, $a > x_i$
s $i = (\ell + d) \operatorname{div} 2$, a onda sve to za izbor

$$i = \left\lceil \frac{\ell + d}{2} \right\rceil.$$

Ima li razlike, kad promijenimo izbor i?

- <u>Zadatak</u>: (Traženje intervala) Zadan je niz $x_1 \leq \ldots \leq x_n$ i $a \in [x_1, x_n]$. Tražim indeks i za koji vrijedi: $a \in [x_i, x_{i+1}]$.

SORTIRANJE NIZOVA

Ključna stvar – sortiramo zato da bismo brže tražili

Primjer – na 1 000 000 podataka:

sekvencijalno – isto 10^6 usporedbi (prosjek pola od toga)

binarno – 21.

Najjednostavniji algoritam – tzv. sortiranje traženjem EKSTREMA.

Recimo da niz $x_1, -, x_n$ želimo PREUREDITI (zamjenom poretka članova) tako da postane uzlazno sortiran

$$x_1 \leq x_2 \leq \ldots \leq x_n$$

Ideja – slično kao sortiranje "kataloških" kartica:

- dovedi najmanji element niza na njegovo mjesto (to je prvo u cijelom nizu)
- ostaje nam dovesti u red skraćeni komad niza $x_2, -, x_n$ (duljine manje za 1, tj. $n-1$)

Još – uoči da je niz duljine 1 već sortiran, pa imamo algoritam:

za $i=1$ do $n-1$ ponavljaj

[u nizu $x_i, -, x_n$ (nesređeni dio), nađi najmanji element (njegov indeks) i onda zamijeni taj element i x_i (tako da najmanji dođe na mjesto i)

– Blokovi: nađi indeks najmanjeg:

$j = i$;

za $k = i+1$ do n ponavljaj

[ako je $x_k < x_j$ onda

[$j = k$

– Zamjena x_j, x_i: ako je $i \neq j$ onda

[temp = x_i

$x_i = x_j$

x_j = temp

Cijeli algoritam:

za $i=1$ do $n-1$ ponavljaj
  $j=i$
  za $k=i+1$ do n ponavljaj
    [ako je $x_k < x_j$ onda
      [$j=k$
  ako je $i<>j$ onda
    [temp $= x_i$
     $x_i = x_j$
     $x_j =$ temp .

Složenost? Što se broji?
Dvije vrste operacija su bitne:
- usporedbe elemenata
- zamjene elemenata

Broj usporedbi: (svaki sa svakim / trenutnim najmanjim)

$$(n-1)+(n-2)+\ldots+2+1=\frac{(n-1)\cdot n}{2}\approx\frac{n^2}{2}$$ (kvadratno u n)

Broj zamjena:
najviše $n-1$ (linearno u n)