ALGORITMOEN DISEINUA
G1: ALGORITMOEN ANALISIA

1) Balantza bat eta txanpon-multzoa ditugu. Haien arteko bat faltsua da. Pisatze kopuruak mugatu kasu onenean, txarrenean eta batez bestekoan:

   a) 10 txanpon baditugu eta faltsua arinagoa bada besteak baino, zenbat pisatze behar ditugu txanpon faltsua mugatzeko?

   b) *n* txanpon baditugu eta faltsua arinagoa bada besteak baino, zenbat pisatze behar ditugu txanpon faltsua mugatzeko?

   c) *n* txanpon baditugu eta faltsua arinagoa ala astunagoa den ez badakigu, zenbat pisatze behar ditugu txanpon faltsua mugatzeko?

   d) Zein metodo erabiliko genuke 24 txanponen artean faltsua 3 pisatze eginez mugatzeko, jakinez honek gehiago pisatzen duela?

2) Suposa dezagun *Alg1* algoritmoak *Konp1* konputagailuan $10^{-4}2^n$ segundo behar dituela n tamaina duen instantzia bat kalkulatzeko.

   a) zenbat denbora beharko du n=10 bada? n=20 bada? Eta n=30 bada?

   b) urtebetez egikaritzen utziko bagenu, gehienez zer tamainako instantzia egikari lezake?

   c) Demagun 100 aldiz azkarragoa den *Konp2* konputagailua erosteko aukera dugula. Zenbat denbora beharko luke *Alg1* algoritmoak *Konp2*an? Urtebetez egikaritzen utziko bagenu, gehienez zer tamainako instantzia ebatziko luke?

3) Algoritmo batek 1000 osagai ordenatzeko segundo 1 behar du zure etxeko konputagailuan. Kalkula ezazu zenbat denbora beharko duen 10.000 osagai ordenatzeko.

   a) algoritmoa n^2 denboraren proportzionala bada.

   b) algoritmoa $n \times lg\ n$ denboraren proportzionala bada.

4)

   a) $t(n)=n^3$ denbora kostua duen algoritmo batekin N tamaina duten problemak ebatz ditzakegu ordu batean. Makina 1000 aldiz azkarragoa balitz, algoritmo bera ordu betez egikaritzen utziez gero, gehienez zein tamainako soluzioak ebatziko genituzke?

b) $t(n)=2^n$ denbora kostua duen algoritmo batekin N tamaina duten problemak ebatz ditzakegu ordu batean. Makina 1000 aldiz azkarragoa balitz, algoritmo bera ordu betez egikaritzen utziez gero, gehienez zein tamainako soluzioak ebatziko genituzke?

5) *n* osagai dituen taula batean x osagaia bilatu egin nahi dugu. Alderaketa kopuruak mugatu kasu onenean, txarrenean eta batez bestekoan:

a) *x* taulan dago eta hau ez dago ordenaturik.

b) *x* taulan ez egotea gerta liteke eta hura ez dago ordenaturik.

c) *x* taulan dago eta hau ordenatuta dago.

d) *x* taulan ez egotea gerta liteke eta hura ordenatuta dago.

e) *x* taulan dago, hau ordenatuta dago eta bertako osagaiak ezin dira sekuentzialki banan bana aztertu.

f) *x* taulan ez egotea gerta liteke, hura ordenatuta dago eta bertako osagaiak ezin dira sekuentzialki banan bana aztertu.

6) Demagun programa jakin bat exekutatzeko PUZko ordu bete daukagula eta ordu horretan makinak kudea dezakeen programaren sarrerarik handiena **n**=1000 000 dela. Kalkulu Zentroko denbora-berrasignatzea ondoren, PUZko 3 ordu esleitu dizkigute. Zein da makinak denbora horretan kudea dezakeen programa beraren sarrerako tamainarik handiena T(*n*) programaren konplexutasuna honakoa bada (k_i konstante batentzat)?

a) $k_1 \times n$.

b) $k_2 \times n^2$.

c) $k_3 \times 10^n$.

7) Eman dezagun gauero programa jakin bat exekutatzeko PUZko ordu bete daukagula eta ordu horretan makinak kudea dezakeen programaren sarrerarik handiena *n*=1.000.000 dela. Orain, gure nagusiak aurrekoa baino 100 aldiz azkarragoa den makina berri bat erosi du. Zein da makina berriak ordu bete batean kudea dezakeen programa beraren sarrerako tamainarik handiena T(*n*) programaren konplexutasuna honakoa bada (k_i konstante ba- tentzat)?

a) $k_1 \times n$.

b) $k_2 \times n^2$.

c) $k_3 \times 10^n$.

8) Jarraiko espresioak egiazkoak ala faltsuak diren arrazoituz adierazi:

a) $2^{n+1} \in O(2^n)$

b) $2^{2n} \in O(2^n)$

c) $n^3 + 2n^2 \in O(n^2)$

d) $3n \in O(2n)$

e) $(n+1)! \in O(n!)$

f) $\log_a n! \in \Theta(n \log_a n)$

9) Ondorengo funtzioak goranzko ordenan zerrenda itzazu. Ordena berdinekoak direnak nolabait hala direla azaldu:

$n^2 + \log n$	$\log (\log n)$	$n^{1+\varepsilon}$ eta $0 < \varepsilon < 1$
2^{n-1}	$\sqrt{n}$	$\ln n$
e^n	$\log_3 n$	$n \log n$
$(\log n)^2$	$\log n^2$	$\frac{n^2}{\log n}$

10) Ondorengo sententzia bakoitzeko froga ezazu egiazkoa den ala ez, bigarrenerako kontrako adibidea emanez.

a) $2^{3n+1} \in O(2^n)$

b) $5 \lg_{10} n \in \Theta(\lg_2 n)$

c) $O(\lg f(n)) = O(\lg g(n)) \Rightarrow O(f(n)) = O(g(n))$

11) Ordena gorakorra jarraituz ordena itzazu ondorengo konplexutasun borneak (hazkuntza tasak), berdinak direnak horrela direla adieraziz: $O(\log n)$, $O(n^n)$, $O(2n^2)$, $O(n!)$, $O(3\ 2^n)$, $O(n \log \log n)$, $O(20)$, $O(n \log n)$, $O(2 \lg n)$, $O(n\ (n+\log n))$, $O(1)$, $O(3^n)$, $O(n\ 2^n)$, $O(n+1)$, $O(n^{n+1})$. Begi-bistakoak ez diren kasuak laburki arrazoi itzazu

12) Ondorengo baieztapenak egiazkoak ala faltsuak diren arrazoi itzazu. Adibidez, "$n^2 \in O(n^3)$" eta "$n^2 \in \Theta(n^3)$" baieztapenei, lehenengoari egiazkoa dela erantzun behar zaio eta faltsua, aldiz, bigarrengoari.

a) $n/\lg n \in \Theta(n)$ eta $n/\lg n \in o(n)$

b) $\lg \lg n \in o(\lg n)$ eta $\lg \lg n \in \omega(\lg n)$

c) $4^{\lg^2 n} \in O(n^2)$ eta $4^{\lg^2 n} \in \Omega(n^2)$

d) $(\lg n)^2 \in o(\sqrt{n})$ eta $\sqrt{n} \in O((\lg n)^2)$

13) Ondorengo errekurtsio ekuazioaren ordena zehatza kalkula ezazu:

t(n)= 8 t(n/2)+ n if n>1;

t(0)= t(1)= 1

Ondorengo multzoren bateko partaidea bada, adieraz ezazu zenena den: $\Theta(n)$, $\Theta(n \lg n)$, $\Theta(n^2)$, $\Theta(n^2 \lg n)$, $\Theta(n^3)$, $\Theta(n^3 \lg n)$, $\Theta(n^4)$.

14) Ondorengo programa zatien T(*n*) eta ordena (ahal baduzu zehatza) kalkulatu.

(a)
```
for IND in LISTEN_FREKUENTZIA'RANGE loop
                        -- Lista 0-z hasieratu
   LISTEN_FREKUENTZIA(IND) := 0;
end loop;
```

(b)
```
IND := 1;              -- Handienaren posizioa aurkitu
for ZENB  in 2..LISTA'LAST loop
   if    LISTA(ZENB)>LISTA(IND)  then IND:= ZENB; end if;
end loop;
```

(c)
```
AURKITUA:= false; IND:= 0; --  Taula  batean  bilaketa  lineala
while IND<LISTA'LENGTH and not AURKITUA loop
IND:= IND +1;
if  ELEMENTUA=LISTA(IND) then AURKITUA := true; end if;
end loop;
```

15) Azter ezazu ondorengo algoritmoa:

```
function Foo (n: in Integer) return Integer is
  X, I:Integer;
begin
  if n<= 1
  then return 1;
  else for I in 1..n loop
         X:= 1;
         while X<n loop
           X:= X*2;
         end loop;
       end loop;
       return (Foo(n/2) + Foo(n/2));
end;
```

16) V(I) ≤ N (*1*≤*I*≤*N*) betetzen duen V(1..N) taula eta X zero ez den digitua emanda, ondoan ematen den algoritmoaren konplexutasuna kalkulatu. Algoritmo honek K aldagaian V taularen elementu positibo guztien X digituaren agerpen kopurua bueltatzen du.

```
for I in V'RANGE loop
   if V(I)>0
   then B:= V(I); K:= 0;
        while B> 0 loop
              if (B rem 10=X) then K:= K + 1; end if;
              B:= B/10;
        end loop;
        Idatzi(V(I),"-an: ", K," aldiz ");
   end if;
end loop;
```

17) BubbleSort, SelectionSort eta InsertionSort algoritmoetan zenbat osagaien mugimendu egiten da kasu txarrenean? Eta zenbat osagai arteko alderaketa? Emaitzak alderatu eta ondorioak atera.

18) BubbleSort eta SelectionSort algoritmoak (eman dizkizuegun metodoak) ez dira sarrerarekiko sentikorrak. InsertionSorta, ordea, bai. Honek, batez bestean egiten dituen alderaketa kopurua kalkula ezazu? Ordena (goi-ordena edo zehatza) hobetzen du batez besteak?

19) Ondorengo programa errekurtsiboen T(*n*)-a eta ordena (O edo Θ) kalkula bedi

```
function FAKTORIALA (N: in INTEGER) return INTEGER is
begin
     if  N=0 then return 1;
     else return     N * FAKTORIALA(N-1); end if;
end FAKTORIALA;


procedure HANOI (N: in INTEGER; I,J:  in Datu123 ) is
-- N diskoen kopurua adierazten du.
begin
     if  N> 0
     then      HANOI (N-1, I, 6-I-J);
               Idatzi(I,"  -> ",J);
               HANOI (N-1, 6-I-J,J);
     end if;
end HANOI ;


function FIBONACCI (N: in INTEGER) return INTEGER is
begin
     if  N=1   then return N;
     else return     FIBONACCI(N-1) + FIBONACCI(N-2); end if;
end FIBONACCI;
```

20) Waste(n) prozedura deiak idazten dituen lerro kopurua jaso beza T(n)-k. T(**n**)-ren ordena zehatza kalkula ezazu

```
Procedure Waste ( x: in INTEGER) is
begin
     for I in 1..x loop
         for J in 1..I loop PUT_LINE(I,J, x); end loop;
     end loop;
     if x>0 then
         for I in 1..4 loop Waste(x div 2); end loop;
     end if;
end Waste;
```

21) Ondorengo algoritmoak sarrerako hitza palindromoa den mugatzen du:

```
function Pal (H:    Hitza; i,j: Indizea) return boolean is
begin
     if i>=j then return true;
     elsif H(i)/=H(j) then return false;
     else return Pal(H, i+1, j-1);
     end if;
end;
```

Aztertu *Pal(Hit, 1, n)* deiak behar duen denbora eta denbora-ordena kasu txarrenean.

22) Ondorengo programen egikaritzapen-denbora ordena aztertu:

```
(a)  function DENA (n:positibo)
     if n=1 then 1 else DENA (n-1) + 2 * PARTZIALA(n-1)
```

jakinik

```
     function PARTZIALA (m:positibo)
     if m=1 then 1 else 2 * PARTZIALA(m-1)
```

```
(b)  function DENA (n,m:positibo)
     if n=1 then m else    m + DENA (n-1, 2 * m)
```

23) `a` bektorea [0..n,0..n] osokoen bektorea izanik, ondorengo eskema azter ezazu. Identifika itzazu sarrera onena (azkarrena) eta txarrena (motelena).

```
I = 1;  J = 1;  kont=0
while I+J ≤ n+1
   if  (a[I-1,J-1] > a[I,J]) { kont++;  I++; }
   else J++
   endif
endwhile
```

24) *n* sarreraren tamaina 3-ren anizkoitza dela suposatuz eta konbinatu funtzioaren ordena konstantea, ondorengo algoritmoaren exekuzio denbora eta ordena kalkula itzazu:

```
programa (a:datuak; n: tamaina): balioa
baldin   n≤3   orduan   zuzenean(a,n)
bestela   m1= n/3
```

```
            m2= n/32
            s1 = programa (a,m1)
            s2 = programa (a,m2)
            itzuli(konbinatu (s1,s2))
endbaldin
```

25) `tratatu` algoritmoa azter ezazu. Sarrera onena (azkarrena) eta txarrena (motelena) zeintzuk diren identifika itzazu

```
kopiatu(a:array[1..n] of integer; i,j:1,…,n)
for k=i+1 to j
    a[k]=a[i]
endfor
end kopiatu

tratatu (a:array[1..n] of integer)
for i=1 to n
   j=i+1
   while  j≤n  and  a[i]≤a[j]
      j++
   endwhile
   kopiatu(a,i,j-1)
endfor
end tratatu
```

26) Honako errekurtsio ekuazioa legokiokeen algoritmo bat idatz ezazu:

$t(n)= 2\ t(n/2) + n \lg n$
$t(0)= t(1)= O(1)$

27) T(n)-ak emanik haien ordenak kalkula bitez:

(a) $T(n)= 2T(n-1)+n+2^n$ $n\geq 1$ $\Theta(n2^n)$
$T(0)= 0$

(b) $T(n)= 2T(n-1)+(n+5)+3^n$ $n\geq 2$ $\Theta(3^n)$
$T(1)= 1$
$T(0)= 0$

(c) $T(n)= 2T(n-1)+3^n$ $n\geq 2$ $\Theta(3^n)$
$T(1)= 1$
$T(0)= 0$

(d) $T(n)= 3T(n-1)+(n+5)+3^n$ $n\geq 1$ $\Theta(n3^n)$
$T(0)= 0$

(c) $T(n)= 3T(n-1)+3^n$ $n\geq 1$ $\Theta(n3^n)$
$T(0)= 0$

28) Jakina da $O(n) \subseteq O(n\lg n) \subseteq O(n\sqrt{n}) \subseteq O(n^2) \subseteq O(n^3)$. Klasifika ezazu O(t(n)) partekotasun kate horretan jakinik: (Lagungarria gerta dakizuke honakoa: $\log_3 4 = 1.26184$)

$$t(n) = \begin{cases} 4t\left(\frac{n}{3}\right) + n & n > 1 \\ 1 & n \leq 1 \end{cases}$$

29) Errekurtsio ekuazioen denbora-ordenak kalkulatu. Hauek ahalik eta sinpleenak eta borne hurbilenak izan behar dute:

a) $t(n) = \begin{cases} t\left(\frac{n}{2}\right) + \mathrm{b} & n > 2 \\ \mathrm{a} & n \leq 2 \end{cases}$ $\Theta(\lg n)$

b) $t(n) = \begin{cases} t\left(\frac{n}{2}\right) + n & n > 2 \\ \mathrm{a} & n \leq 2 \end{cases}$ $\Theta(n)$

c) $t(n) = \begin{cases} 2t\left(\frac{n}{2}\right) + 2n & n > 2 \\ \mathrm{a} & n \leq 2 \end{cases}$ $\Theta(n \lg n)$

d) $t(n) = \begin{cases} t\left(\frac{n}{2}\right) + \lg n & n > 2 \\ \mathrm{a} & n \leq 2 \end{cases}$ $\Theta((\lg n)^2)$

e) $t(n) = \begin{cases} 4t\left(\frac{n}{2}\right) + n & n > 2 \\ \mathrm{a} & n \leq 2 \end{cases}$ $\Theta(n^2)$

f) $t(n) = \begin{cases} 4t\left(\frac{n}{3}\right) + 3n - 5 & n > 3 \\ 1 & n \leq 3 \end{cases}$ $\Theta(n^{1.26})$

g) $t(n) = \begin{cases} 2t\left(\frac{n}{4}\right) + \lg n & n > 4 \\ 1 & n \leq 4 \end{cases}$ $\Theta(\sqrt{n})$

h) $t(n) = \begin{cases} t\left(\frac{n}{4}\right) + \sqrt{n} - 1 & n > 4 \\ 1 & n \leq 4 \end{cases}$ $\Theta(\sqrt{n})$

i) $t(n) = \begin{cases} 8t\left(\frac{n}{2}\right) + n & n > 2 \\ 1 & n \leq 2 \end{cases}$ $\Theta(n^3)$

j) $t(n) = \begin{cases} 2t\left(\frac{n}{4}\right) + \sqrt{n} & n > 4 \\ 1 & n \leq 4 \end{cases}$ $\Theta(\sqrt{n} \lg n)$

k) $t(n) = \begin{cases} 2t\left(\frac{n}{2}\right) + \sqrt{n} & n > 2 \\ 1 & n \leq 2 \end{cases}$ $\Theta(n)$

30) Testu editore askok ondorengo algoritmoa erabiltzen du. String konkretu baten lehenengo agerraldia kalkulatzen du (hau da, B(1..m) karaktere-zerrendaren lehenengo agerraldia A(1..n) karaktere-zerrendan bilatzen du) B stringa A-n hasten den indizea itzuliaz, Barne dagoenean noski. *Muga = n − m + 1* balioa B stringa A-n has litekeen eskuinaldeagoko indizea da.

```
procedure StringSearch (A,B: in String; Aurkitua: out boolean;
```

```
                              Hasiera: out Indizea) is
      N:= A'Length; Topatua:= false; M:= B'Length; Muga:= n-m+1;
      I, J : Indizea; Hasi:= A'First;
begin
      while not Topatua and (Hasi /= Muga) loop
          I:= Hasi;
          J:= B'First;
          while J/= M+1 and then (A(i)=B(j)) loop
                I:= I+1; J:=J+1;
          end loop;
          Topatua:= (J=M+1);
          if not Topatua  then  Hasi:= Hasi+1;  end if;
      end loop;
      Aurkitua:= Topatua;
      Hasiera:= Hasi;
end StringSearch;
```

Kasu txarrenean zenbat aldiz egikaritzen da A(i) = B(j) alderaketa? Zeintzuk sarrerak dira kasu txarrena ematen dutenak? Ohar zaitez Adan testa bakarrik J≠M+1 baliozkoa denean kalkulatzen dela.

31) `Mokokatzen` izeneko algoritmoak V[a...b] bektoreko hainbat balioen batura kalkulatzen du. Algoritmoaren exekuzio ordena kalkula ezazu:

```
funtzioa Mokokatzen (V,a,b) return zenbakia
X ← ⌊(a+b)⌋/2
if (b-a+1)≤3 then return V(X)
else Y ← ⌊(2a+b)⌋/3
     Z ← ⌈(a+2b)⌉ /3
     return V(X) + Mokokatzen (V,a,Y) + Mokokatzen (V,Z,b)
```

Idazkera asintotikoek definitutako o(n), $O(\sqrt{n})$, $\Omega(\sqrt{n})$, $\Omega(2^n)$ edo Ω(n lg n) multzoetako zein multzokoa da lortu duzun kostu funtzioa?

32) Izan bedi P(*n*) n mailako polinomioa: $P(x) = a_nX^n+a_{n-1}X^{n-1}+ \ldots +a_1X^1+a_0$ Bere koefizienteak *Koef* izeneko taula batean metatu dira, ondorengo moduan: *Koef*(i) = a_i X aldagaiaren balioa emanik, polinomioaren balioa kalkulatuko duen programa definitu nahi dugu:

```
Koef: array (0..N) of Zenbakia := HASIERATU_K;
X: Zenbakia:= HASIERATU_X;
function POLIEBAL (M: 0..N) return Zenbakia is
begin
     if M=0 then return Koef(0);
     else      Ber:= 1;
               for Ind in 1..M loop Ber:=Ber * X; end loop;
               return (Koef(M) * Ber)+ POLIEBAL (M-1);
     end if;
end POLIEBAL;
```

a) Jaso beza T(n)-k POLI EBAL(n) deiak egiten dituen eragiketa aritmetiko (+,−,∗,/) kopurua. Kalkula ezazu T(n) exekuzio-denbora.

b) Aurreko ataleko kopurua errez hobe daiteke. Honela, eta ** berreketaren eragile estandarra erabili gabe, emaitza berdina ekoiztuko duen eta T(n) hobea duen algoritmoa idatzi eta haren exekuzio-denbora ekuazio berria txikiagoa dela froga ezazu.

33) Ondorengo funtzio errekurtsiboak X arruntaren M. berredura kalkulatzen du:

```
X: Oinarria := HASIERATU_X;
function BER (X,M: Arrunta) return Arrunta is
begin
     if M=0 then return 1;
     elsif M=1 then return X;
     elsif M mod 2 = 0
        then return BER(X, M/2) * BER(X,M/2);
        else return X* BER(X,M/2) * BER(X,M/2); end if;
end BER;
```

a) Algoritmoak egikaritzen duen biderketa kopuru konkretua kalkulatu, bai eta ordena zehatza ere.

b) Aurreko ataleko algoritmoak kalkulu berdinak errepikatzen dituela ikusiz, balio bera kalkulatzen duen algoritmo errekurtsibo berria idaztea eskatzen da, baina berreketa (**) eragiketa estandarra erabili gabe. Bertsio berriak biderketa kopuru gutxiago egin beharko du. Algoritmo berriaren ordena hobea izan behar du, eta hala dela frogatu.

ALGORITMOEN DISEINUA

G2: DATU-EGITURA AURRERATUAK:
meta, partiketa eta grafoa

1) Izan bedi G grafo zuzendua. G auzokide listen bidezko adierazpidea erabiliz emana dator. Idatzi eta aztertu

   a) G-ren adierazpidea auzokidetza matrizera bihurtzen duen metodoa eta aztertu.

   b) G-ren adierazpidea inzidentzi matrizera bihurtzen duen metodoa eta aztertu.

2) Izan bedi G grafo zuzendua. G auzokidetza matrize bidezko adierazpidea erabiliz emana dator. Idatzi eta aztertu

   a) G-ren adierazpidea auzokideen listara bihurtzen duen metodoa eta aztertu.

   b) G-ren adierazpidea inzidentzi matrizera bihurtzen duen metodoa eta aztertu.

3) $G=(E,A)$ grafo zuzendu baten karratua $G^2=(E,B)$ da, non $(u, w)\in B$ arkuak existitzen duen baldin eta soilik baldin $v\in E$ erpin batentzat existitzen badute $(u, v) \in A$ eta $(v, w) \in A$ arkuek. Hots, G^2-k arku bat du u-tik w-ra, G-k zehazki u-tik w-raino 2 luzerako bidea duenean. Idatzi algoritmo eraginkorrak G^2 kalkula dezaten:

   a) G auzokide listen bidez adierazirik dagoenean, eta

   b) G auzokide matrize bidez adierazirik dagoenean.

a) G grafo zuzendua emanik eta bertako i erpina, *Indegree*(i) zenbakia kalkulatu nahi dugu; hots, i auzokide duten erpinen kopurua. Grafoen ohiko bi adierazpideetarako algoritmo bana idatzi *Indegree*(i)a kalkulatzeko, hauen denbora kostuak mugatu eta argudia ezazu bi adierazpideetatik zein den egokiena.

4) G=(E,A) grafoa auzkokideen zerrendak erabiliz definiturik dagoena, irudika ezazu:

V=[V(1),V(2),V(3),V(4),V(5),V(6),V(7),V(8)]	
V(1)=[4]	V(5)=[1,2,6]
V(2)=[1]	V(6)=[2,7]
V(3)=[8]	V(7)=[3]
V(4)=[2,3,8]	V(8)=[7]

   a) 1 erpinetik hasiz Sakoneran korritzerakoan sortzen den zuhaitza marraztu, dagoeneko bisitatutako erpinak puntuzko marraz lotuz. 1 erpinetik hasi den korritze

horretako erpinen eta arkuen zerrenda idatzi lehenengo bisita ordenan. (a,b) idazkera erabil ezazu a-tik b-ra doan arkua adierazteko)

b) 1 erpinetik hasiz zabaleran korritzerakoan sortzen den zuhaitza marraztu, dagoeneko bisitatutako erpinak puntuzko marraz lotuz. 1 erpinetik hasi den korritze horretako erpinen eta arkuen zerrenda idatzi lehenengo bisita ordenan.

5) <2, 16, 7, 4, 18, 30, 52, 46> osokoen sekuentzia emanik meta-eraiki: (a) *Azaleratu* erabiliz eta (b) *Hondoratu* erabiliz

6) <10,2,9,16,8,6,1,3,12> taula erabiliz meta bat (*Heap*) eraiki ezazu soilik hondoratu prozedura erabiliz. Metaren egoera marraztu behar duzu (bai zuhaitz bitarra bezala bai eta honek adierazten duen taula bezala) eraiketa-pauso garrantzitsu bakoitzaren ondoren.

7) *n* osagai desberdin dituen bektore baten adibide konkretua eman ezazu, non *M* eta gaineko honako agindu sekuentzia aplikatu ondoren meta desberdina lortuko dugun.

```
M:=MetakoHandiena(B(1..n));
ErroaEzabatu(B);
OsagaiaGehitu(B(1..n-1), M);
```

8) Demagun 12 osagai dituen *B(1..12)* bektorea dugula, non *B(i)=i* gertatzen den $i \leq 12$ *i*-ren balio guztientzat. Jarraiko agindu sekuentzia egikaritu ondoren bektorearen egoera marraz ezazu. Aginduak bata bestearen ondoren exekutatzen dira, eta lehengoa ezik, gainontzekoak aurreko aginduak itzuli duen bektore gainean egikarituko da.

```
MetaEraiki(B);
EguneratuMetaBalioz(B,12,10);
EguneratuMetaBalioz(B,1,6);
EguneratuMetaBalioz(B,5,8);
```

9) Ondorengo galderei erantzun:

a) *h* altuera duen meta batek, gutxienez zenbat osagai eduki ditzake? Eta gehienez?

b) *Min-Meta* oro txikienetik handienera ordenaturik dagoen bektorea da? Eta alderantziz?

c) *Max-Meta*-k eraikitzeko Hondoratu prozedura erabiliz egin liteke. Metodo honek zergatik `n/2..1` posizioak aipaturiko ordenan hondoratzen ditu eta ez `1..n/2` ordenan?

10) Metatik_Ezabatu(A,i) izeneko algoritmoa egizu, A max-metako i posizioko osagaia metatik kenduko duena. Algoritmoaren denbora ordena O(lg *n*) izan behar du, max-metak *n* osagai dituenean.

11) *k* zerrenda ordenatuak, *n* osagaiko zerrenda ordenatu bakarrean biltzen dituen algoritmoa egizu (hots, *k* zerrendetan dauden osagaien kopurua guztira *n* da). Algoritmoaren exekuzio denbora O(n lg *k*). (Iruzkina: min-meta egitura bateraketa-bilketa egiteko erabiltzea gomendatzen zaizu).

12) `k-meta` funtsean beterik dagoen *k*-adarretako zuhaitzak erabiliz osatutako meta bezala defini dezakegu:

   a) `k-meta` bektore batean eraginkorki nola biltegi litekeen deskriba ezazu, azalduz zeintzuk eragiketa egin behar diren edozein erpin baten gurasoa eta umeak lortzeko.

   b) `k-meta` -ko `Hondoratu` eta `Azaleratu` eragiketen kostuak aztertu.

   c) Metak maneiatzen dituzten gainontzeko eragiketetan (`Max-Meta-Eraiki(V)`, `MetariGehitu(V,B)`, `MetatikKendu(V,p)`, `EguneratuMeta(V,p,B`)) egin behar diren aldaketak zehaztu, `k-meta` berrira egokituak izan daitezen.

   Metak erabiltzen dituen ordenazio algoritmoa berriz ere azter ezazu, oraingoan `k-meta` darabilkien inplementazioa duen bertsioa duzula suposatuz.

13) Izan bitez `T1` eta `T2` bi meta, zehazki biek 2^n osagai dituztenak. Hauetan dauden osagaiak konbinatuko dituen prozedura bat aurki ezazu `T3` zuhaitza eraikitzeko, prozedurak egin ditzakeen osagaien arteko konparaketa ordena `o(n)` izan behar du. (Oharra: prozeduraren kostu ordena handiagoa izan badaiteke ere, metetako elementuen arteko eragiketa kopurua `o(n`)koa izan behar du).

14) Ondorengo ariketa `Heapsort` algoritmoaren bertsio berri bat aztertzen du. Hain zuzen, bertako metaren eraiketaren prozesua aldatzen da.

```
procedure HeapSort (S: in out Sek) is
begin
     MetaEraiki2(S'First, S'Last);
     for Heaparen_Luzera   in reverse  S'First+1 .. S'Last loop
         STaulanTrukea(S'First, Heaparen_Luzera);
         ErroaHondoratu(S'First, Heaparen_Luzera -1);
     end loop;
end HeapSort;
```

15) Meta datu-mota abstraktua abiapuntutzat harturik, taula bat emanik hura metan bilakatuko duen `MetaEraiki2` algoritmoa idaztea eskatzen da.

Baina, haren eraiketarako *AzkenaAzaleratu* eragiketaz bakarrik balia daiteke programatzailea. *AzkenaAzaleratu* prozeduraren zehaztapena ondorengoa da:

```
procedure AzkenaAzaleratu (Hasi,Buka: in SekIndizea);
Sarrera:  Hasi eta Buka-1 indizeek S taulan meta bat mugatzen
          dute.
Buka indizean dagoen balioa S-en azaleratu nahi den osagaia da.
Irteera:  Hasi eta Buka-1 indizeek S taulan meta bat mugatzen
          dute.
Efektua:  S(Hasi,Buka) maximoen metan, metaren propietatea
          berreskuratu arte S(Buka) balioa azaleratzen du.
```

a) *AzkenaAzaleratu* prozeduraren kodea honako hau bada:

```
procedure AzkenaAzaleratu (Hasi,Buka: in SekIndizea) is
  Ind_J: SekIndizea;Ind_K: SekIndizea:=Buka;
begin
  loop
    Ind_J := Ind_K;
    if  Hasi<Ind_J  and then  S(Ind_J/2)<S(Ind_K)
    then  Ind_K:= Ind_J/2;  end if;
    STaulanTrukea (Ind_J, Ind_K);
    exit when Ind_J = Ind_K;
  end loop;
end AzkenaAzaleratu;
```

Egindako lana neurtzeko eragiketa adierazgarri bezala S taulako osagaien arteko alderaketak hartzen baditugu (non S taulak hasieran n osagai dituen), `MetaEraiki2` kasu txarrenean zein ordena zehatzekoa da? `HeapSorten` zein bertsio da azkarragoa: aurreko atalekoa edo Hondoratu prozedura erabiliz lineala dena? Erantzuna arrazoitu bedi gehienez 5 lerro erabiliz.

16) (a) Taula meta bihurtzen duen ondorengo algoritmoa aztertu:

```
procedure MetaEraiki (V(1..m)) is
begin
  for J in reverse 1 .. m div 2 loop
    Hondoratu(V(1..m), J);    -- V(J) posizioko balioa hondoratu
  end loop;
end;
```

(b) Aurreko ataleko ordena O(*m*) lortu ondoren, non dago akatsa HeapSort aztertzerakoan honako arrazonamenduan?

• O(*n*) denboran *MetaEraiki(T(1..n))* algoritmoak $\left\lfloor \frac{n}{2} \right\rfloor$-ren aldiz egikaritzen du Hondoratu.

• Beste algoritmo honek:

```
procedure HeapSort (T(1..n)) is
begin
  MetaEraiki(T(1..n));
  for J in reverse 2 .. n loop
    Trukea(T(1),T(J));
    Hondoratu(T(1..J-1), 1);  -- Erroa hondoratu
  end loop;
end;
```

Meta eraikitzen du lehengo ataleko algoritmoa erabiliz. Ondoren, begiztak *n−1* aldiz egikaritzen du (*Trukea*+*Hondoratu*); edo ordena aldetik berdina dena, $2(\lfloor \frac{n}{2} \rfloor$x *Trukea*+*Hondoratu))*

Aurreko bi puntuetako hiru zatiek O(*n*) ordena dutenez, `HeapSort` algoritmoaren ordena O(*n*) da.

17) Demagun *n* osagai dituen taula bateko *K* giltza handienak kalkulatzeko honako algoritmoa ematen digutela:

```
function  KGiltzaHandienak  (V: in Taula; K: in Integer)
                                  return Taula is
  M: Taula; Giltzak: Taula(1..K);
begin
  MMetaEraikiVEmanik(V,M);
  for  J  in 1 .. K  loop
    Giltzak(J):= M(1);
    M(1):= M(N-J+1);           -- Azkena errora
    ErroaHondoratu(M(1..n-J));
  end loop;
end KGiltzaHandienak;
```

Algoritmoaren ordena lineala, O(*n*), izan dadin, zer ordena eduki behar du *K*-k (*n*-ren funtziopean)?

18) *V* taulak *n* zenbaki arrunt desberdin du. V-ko *m* osagai txikienak kalkulatu nahi ditugu. Bestalde, *m*<<<*n* betetzen dela badakigu. *m* zenbakien kalkulua modurik eraginkorrenean egin nahi dela eta, nola kalkulatuko zenituzke?

a) *V* ordena gorakorra jarraituz sailkatu eta lehenengo *m* osagaiak itzuliz?

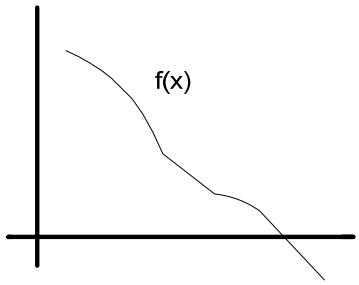
b) *KHautaketa(V,k)*ri m aldiz deituz (non k=1,2, ... eta m)?

c) Beste metodoren bat erabiliz? Zein?

Zure erantzuna argudia ezazu kasu txarrenetako eta batez besteko kasuetako analisi ezagunak alderatuz.

19) Ondorengo algoritmoa osatu n osagaien arteko K garren handiena kalkula dezan. n osagaiak banaka lortu behar dira *get-number(X : out integer)* prozedurari n dei eginaz. Kasu honetan erabilitako espazio estra minimizatu behar da. Hori dela eta, beti $\Omega(n)$ memoria- espazio estra behar duten soluzioak ez dira onartuko.

```
for  I  in  1 .. n  loop
   get-number (X)
   …
   end loop;
return KGarrena;
```



20) G=(ERP,ERT) grafo ez-zuzendu baten osagai konexuak kalkulatzeko partiketa egitura erabiltzen duen ondorengo algoritmoa ematen da:

```
proz  OsagaiKonexuak (G)
-- Partiketaren hasieraketa
hasi
   for  Da(a, ERP erpin bakoitzarentzat)  egin
      Multzoa_egin(v)
   end for;
   for  Da((x,y), ERT ertz bakoitzarentzat) egin
      if Bilatu(x) <> Bilatu(y)
      then Bateratu(Etiketa(x),Etiketa(y))
      end baldin;
   end for;
end proz;
```

Grafoak *K* osagai konexu baditu, zenbat Bilatu eragiketa egingo da? Eta zenbat *Bateratu* eragiketa? Emaitza $|ERT|$, $|ERP|$ eta *K*-ren funtziopean eman.

ALGORITMOEN DISEINUA

G3: GRAFO GAINEKO ALGORITMOAK:
sakonerako eta zabalerako korritzeak

1) Ondorengo auzokide-zerrendak G grafoa zuzendua adierazten du:

```
E = [E(1), E(2), E(3), E(4), E(5), E(6), E(7)]
E(1) = [2, 3, 6]
E(2) = [3, 4]
E(3) = []
E(4) = [1, 3]
E(5) = [3, 7]
E(6) = [1, 3]
E(7) = [4, 5]
```

Marraz ezazu adierazpide honi dagokion grafoa. G grafoan 1 erpinetik hasiz sakonerako korritzeari dagokion erpinen zerrenda -lehenengo ikustaldi ordenan- idatz ezazu. G grafoan 1 erpinetik hasiz sakonerako korritzeari dagokion arkuen zerrenda - lehenengo ikustaldi ordenan - idatz ezazu. (a erpinetik hasi eta b erpinean amaitzen den arkua adierazteko (a, b) idazkera erabil ezazu.).

Ondoren, kalkulu berdinak egin zabalerako korritzerako.

2) Izan bedi G grafo zuzendua. G auzokidetza matrize bidezko adierazpidea erabiliz emana dator. Idatzi eta aztertu

   a) G-ren sakonerako algoritmoa adierazpide berrirako egokitu eta aztertu.

   b) G-ren berdin, baina zabalerako korritze algoritmorako.

3) Grafoen gaineko sakonerako korritzea idatzi sarrerako grafoa auzokide matrize bidez adierazirik datorren kasurako (grafoaren adierazpidea beste batetara bihurtu gabe). Bere ordena aztertu. Hobea al litzateke auzokideen lista adierazpidera bihurtzea eta orduan adierazpide berrian sakoneran korritzea? Erantzuna argudia ezazu.

4) Grafoen gaineko zabalerako korritzea idatzi sarrerako grafoa auzokideen listen bidez adierazirik datorren kasurako (grafoaren adierazpidea beste batetara bihurtu gabe). Bere ordena aztertu. Hobea al litzateke auzokide matrize adierazpidera bihurtzea eta orduan adierazpide berrian zabaleran korritzea? Erantzuna argudia ezazu.

5) `G=(Erpinak, Arkuak)` grafo zuzenduentzat beste adierazpide berezi bat `|Erpinak|×|Arkuak|` intzidentzia matrize bidezko adierazpidea da. *IM* intzidentzia matrizeko IM_{ij} posizioan ager daitekeen balioa

$$IM_{ij} = \begin{cases} +1 & j \text{ arkua } i \text{ erpinetik irtetzen da} \\ -1 & j \text{ arkua } i \text{ erpinara iristen da} \\ 0 & \text{bestela} \end{cases}$$

izan daiteke. Adierazpide konkretu hau duen grafo bat sakoneran korritzen duen algoritmoa idatzi bedi. Algoritmoaren ordena zehatza zein da? Algoritmoak egindako lana aztertuz argudia bedi.

Zabalerako korritzearekin gauza bera egin bedi.

6) Z zuhaitza emanik bertako zein K mailak adabegi gehien dituen kalkulatuko duen algoritmoa idaztea eta bere ordena kalkulatzea eskatzen da. Horietako maila bat baino gehiago existituko balira, maila handiena itzuli beharko luke algoritmoak.

7) Auzokideen listen bidez adierazitako grafo zuzendu bat emanik, zenbat denbora beharko litzateke erpin bakoitzeko d^+, `outdegree`, kalkulatzeko? Eta d^-, `indegree`, kalkulatzeko?

8) Z abalerako edo sakonerako korritzea behar den neurrian aldatu Ordenazio Topologikoaren problema ebatzi dezan. Soluzioaren ordena zein da?

9) *n* erpin eta *a* arku dituen grafo zuzendu azikliko bateko erpinei [1..n] tarteko zenbaki desberdinak esleitzen dien algoritmo bat idaztea eskatzen da, asignazio metodoak ondorengoa izan behar duelarik:$\forall v, w$ grafoko erpinak izanik,

`baldin` `v ≠ w` **`eta baldin`** `v-tik w-raino bidea existitzen badu`
`orduan` `Zenbakia(v)<Zenbakia(w)`

Algoritmoaren ordena zehatza zein da? Erantzun biak argudia bitez.

10) *n* konputagailuz osaturiko sare internazional bateko X konputagailu bakoitza dobloi bat ordainduz bere I konputagailu auzokidearekin komunika daiteke. Aldi berean, I beste batzuekin komunikatuta dagoenez, X beste konputagailuekin komunika daiteke. Noski, komunikatzen den konputagailu bakoitzari dagokion dobloia ordaindu beharko dio.

`Prezioa(1..N)` taula kalkulatuko duen algoritmoa idatz ezazu, non `Prezioa(J)` gelaxkak 1 konputagailua J konputagailuarekin konexioa eduki dezan, lehenengoak ordaindu beharko duen dobloi kopurua jasotzen duen.

11) Matrize bidezko adierazpidea erabiltzen denean grafo gehienek $\Theta(|\texttt{Erpinak}|^2)$ denbora behar dute. Halere, salbuespen batzuk badaude. Bestalde, grafo bateko erpin bat isuri-toki erpina da, baldin eta $d^-=|\texttt{Erpinak}|-1$ eta $d^+=0$ baditu. Froga bedi grafo batek isuri-toki erpinik duen edo ez mugatzeko $O(|\texttt{Erpinak}|)$ denbora behar duen algoritmo batekin aski dela, nahiz grafoa auzokide matrize bidez adierazirik eman.

12) Demagun bidai-agentzia baten datu-basean munduko aireportu guztien arteko hegaldiei buruzko informazioa gordetzen dela. Aire-lotura guztiak joan eta etortzekoak direla suposatuko dugu. Ondoko problema ebatzi nahi dugu: A eta B aireportuak emanda, zein da Atik Bra hegazkinez joateko geldialdi kopuru minimoa?

    a) Grafoa korritzeko algoritmo ezagunetatik, zein uste duzu egokiena dela? Zergatik? Aukeratutako algoritmoari egin beharreko aldaketak azal itzazu.

    b) Ezagutzen dituzun grafoei buruzko algoritmoen artean, problema hau ebazten duten haietako bakoitzaren denbora-konplexutasuna esan ezazu

    c) Aurreko bi ataletan aipatutakoen artean, zeinekin gelditu zinateke azkenean? Zergatik?

13) `G=(Erpinak, Ertzak)` grafoa zuhaitza ote den mugatzen duen algoritmoa idatzi. Grafoa konexua balitz, algoritmo berdina erabiliko zenuke? Horrela ez balitz, aurrebaldintza bezala grafo konexua eskatuko lukeen algoritmoa idatzi.

14) Grafo batek ziklorik ote duen mugatuko duen algoritmoa idaztea eskatzen da

    a) grafoa ez-zuzendu denean, eta

    b) grafoa zuzendu denean.

    Aurreko ataletan sakonerako korritzea erabili baduzu, berridatz itzazu algoritmoak zabalerako korritzea erabiliz problema berdinak ebatzi ditzaten, eta alderantziz.

    Zein estrategia nahiago duzu? Arrazoi sendorik baduzu horretarako? Zein?

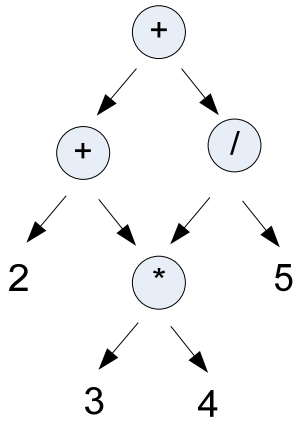
15) (X_i, Y_i) jatorri puntutik (X_n, Y_n) helmuga puntura zenbat bide posible existitzen den mugatzen duen algoritmo bat idatzi bedi. Puntuen koordenadak zenbaki arruntak direla suposatu. Bestalde, (X, Y) puntu batetik beste puntu batera joateko eman daitezkeen pausoak (X+1, Y) eta (X, Y +1) bakarrik izango dira. Algoritmoaren exekuzio-denbora aztertu.

16) Jakina da bide oro Erromara doala. Gida on bat erabiliz eta errepide mapa bat erabiliz, DAG bat marraztu dugu (grafo zuzendu aziklikoa) hamaika ibilbide on jasotzen dituena gure hiritik Erromara iristeko. Emandako DAGa erabiliz, gure hiritik Erromara dauden ibilbide kopurua kalkulatuko duen metodoa diseinatzea eskatzen zaizu bai eta haren denbora-analisia kalkulatzea ere.

17) *Esfortzulogia* Fakultateko Ikasketa Planak N irakasgai ditu. Irakasgaiak elkarren artean goi-baldintza bitartez erlazionatuta daude: A irakasgaia B irakasgaiaren goi-baldintza bada, orduan ikasleak B-n matrikula egin ahal izateko ikasleak A irakasgaia aurreko urtean gainditua izan beharko du. Ikasketa planean hasierako zenbait irakasgai badago (hots, goi-baldintzak ez dituena), bai eta bukaerako zenbait irakasgai ere (hots, beste irakasgaien goi-baldintza ez dena); eta , begi bistakoa den bezala, ez dago goi-baldintzen kate zirkularrik (ezinezkoak bailirateke bertako irakasgaiak egitea). Ikasleak nahi haina irakasgai aldi berean egiteko libre dira, noski, goi-baldintzak errespetatzen dituzten bitartean. Irakasgai guztiak derrigorrezkoak dira eta uneren batean gainditu behar dituzte titulu eskuratu nahi badute.

DEA irakasgaia behin eta berriro ez gainditzeaz nazkatuta zaude eta Esfortzulogiara joatea pentsatzen ari zara, bertako irakasgaiak esfortzurik gabe gainditzen baitira eta lanbide-irteera handikoa baita. Baina merezi ote dizun ikusi nahi duzu. Zehazki, jakin nahi duzuna titulua lortzeko beharko duzun urte kopuru minimoa da. Bila ezazu algoritmo eraginkor bat kopuru minimo hori kalkulatzeko. Problema ebazteko erabili duzun teknika xehetasunez azal ezazu.

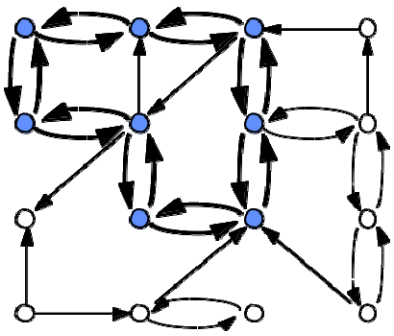
18) Espresio aritmetikoa DAGen (grafo zuzendu aziklikoen) bidez adieraz daiteke. Adierazpen hau zuhaitzen bidezkoa baino trinkoagoa da, errepikatutako azpi-espresioak behin bakarrik agertzen dira eta. Adibidez, irudiko DAGak 2 + 3 * 4 + (3 * 4)/5 espresio aritmetikoa adierazten du. Eragile (edo barne adabegi) bakoitzak eragigai (edo arku) bi ditu zehazki. Ezaugarri hauek dituen DAGa ebaluatuko duen algoritmo lineal bat adabegien kopuruan idatzi. Bere denbora-ordena ere kalkula ezazu.



19) Grafo bat *birpartitua* da baldin eta soilik baldin bere erpinen multzoa bi azpimultzo disjuntutan partitu badaiteke non grafoko ertz bakoitzeko erpinak multzo desberdinetan dauden.

Bila ezazu grafo konexu eta ez zuzendu bat *bipartigarria* den ala ez mugatzen duen eta ertzen kopuruan lineala izango den algoritmo bat.

20) Grafo zuzendu bat emanik, *eraztuna* deritzogu hiru erpin edo gehiago ziklo moduan eta norantza bietan konektatuta dituen serieari. Adibidez, irudiko grafo zuzenduan:



beltzez markaturik dago grafo honek duen eraztun bakarra. Sakonerako korritzean inspi- ratuz, diseina ezazu algoritmo eraginkor bat grafo zuzendu bat emanik honek gutxienez eraztun bat duen ala ez kalkulatuko duena. Zure soluzioaren denbora-ordena kalkula ezazu.

21) Zuhaitz baten *diametroa* edozein erpin bikote arteko distantzia guztietatik distantzia handiena da. Zuhaitz baten diametroa kalkulatzen duen algoritmo lineal bat eraikitzea eskatzen da.

22) G=(N,A) grafo ez-zuzenduaren G'=(N',A') azpigrafoa osagai konexu bat dela esaten da, baldin eta soilik baldin G' konexu bada eta G-ren G"=(N",A"") azpigrafo konexuak existitzen ez badu N'⊂N" edo A'⊂A" betez (hau da, azpigrafo konexu maximala da).

Grafo baten osagai konexu guztiak adierazteko grafoen gaineko zabalerako korritzea erabil ezazu. OK(1,…,n) egitura eskuragarri duzu osagai konexuak metatzeko, bertako OK(i) osagaiak osagai konexu bat biltegi baitezake. Gainera osagai konexu bakoitzeko adierazpidea ekoizteko ondorengo eragiketak erabil ditzakezu:

- Sortu-Grafoa(G): erpinik eta ertzik gabeko G grafoaren hasieraketa
- Erpina-Gehitu(G,i): i erpina G grafoari gehitu
- Auzokidea-Gehitu(G,i,j): G grafoko i erpinari j erpin auzokidea gehitzen zaio (oharra, i erpinak G grafokoa izan behar du eragiketa aplikatu ahal izateko).

23) Grafo konexu ez-zuzendu eta pisudun bat dugu. Grafoko k arku arinenak eraginkorki mugatuko dituen algoritmo bat egizu, jakinik $k \in o(a)$ dela (a grafoko ertz kopurua izanik) eta gehienez erabilgarri duzun espazio estra o(a) dela (hau da, espazio estrak ezin dun a–n lineala izan). Zure soluzioa arrazoitu ondoren, aztertu haren denbora ordena eta memoria espazio estra.

24) *Baso* bat zuhaitzez osatutako grafo bat da. G=(N,A) grafo ez zuzendu bat emanik, G grafoa basoa den ala ez erabakitzen duen algoritmoa egizu. Grafoa basoa denean, basoko zuhaitz kopurua itzuli behar du algoritmoak. Algoritmoaren analisia egizu.

25) Prozesadore bakarra izanik n ataza egin behar ditugu. Ataza batzuk beste batzuen menpe daude; hau da, ataza baten aurrekari guztiak amaitu ez diren bitartean, bera, menpekoa, ezin egin liteke. Adibidez, {1,2,3,4,5} atazak izan ditzakegu eta honako menpetasun informazioa: 2 ataza 1en menpekoa da, eta 3,4 atazak ere 1en menpekoak, aldiz 5 ataza 3 atazaren menpe dago.

Planifikazio egingarri bat atazen permutazio bat da, non atazak permutazioan jasotako ordenan egikari litezkeen eta ataza bat hasten denean, honen aurrekari guztiak dagoeneko amaituak dauden.

a) Planifikazio egingarri bat kalkulatuko duen algoritmo bat egizu, baldin eta existitzen badu. Zein da zure soluzioren ordena? Aurreko adibidean egingarriak lirateke: [1,3,5,2,4], [1,2,3,4,5],… baina ez [1,5,2,3,4]

b) Adibide batez azaldu ezazu planifikazio egingarririk ez duen egoera bat.

c) Demagun i ataza bakoitzak di iraupena esleitua duela. Adibidez, d1=5, d2=10, d3=3, d4=7, d5=4. Denbora minimoan ataza guztiak burutzeko algoritmo bat idatz ezazu, suposatuz nahi adina prozesadore dituzula. Zein da zure soluzioaren konplexutasun ordena?

ALGORITMOEN DISEINUA
G4: ZATITU ETA IRABAZI TEKNIKA

1) Problema jakin bat ebazteko, algoritmo sinple bat daukagu. Honek, *n* tamainako sarrerak ebazteko koadratikoa den denbora-ordena hartzen du (hau da, $t(n) \in O(n^2)$). Problema bera ebazten duen eta *zatitu eta irabazi* teknika jarraitzen duen beste algoritmo bat aurkitu da ordea. Estrategia hori jarraituz, *n lg n* eragiketa egiten da problema tamaina erdiko bi azpiproblematan banatzeko. Aldi berean, beste *n lg n* eragiketa egin behar da azpiproblemen emaitzak konbinatuz jatorrizko problemaren emaitza lortzeko. Zatitu eta irabazi algoritmo berria hasierako algoritmoa baino eraginkorragoa da?

2) Ondorengo algoritmoak osokoen bektore bat emanik, bertako osagai handiena eta txikiena mugatzen ditu. Kasu txarrenean algoritmoak osagaien arteko zenbat alderaketa egiten dituen kalkula bedi.

```
proz MaxMin (T[i..j], Max, Min)
-- i <= j
begin
  if T[i]<=T[j] then Max:= T[j]; Min:= T[i];
  else  Max:= T[i]; Min:= T[j]; end if;

  if i+1<= j-1
  then  MaxMin (T[i+1..j-1], Lag_Max, Lag_Min);
        if Max<Lag_Max then Max:= Lag_Max; end if;
        if Lag_Min<Min then Min:= Lag_Min; end if;
  end baldin;
end proz;
```

3) Ondorengo algoritmoa telefono-zerrenda batean abizenak bilatzeko balio du. (Noski, zerrenda alfabetikoki ordenaturik dago.)

```
fuction Bilatu (Zr: in Zerrenda; Abizena: in String;
                Salto: in Positive) return Integer is
-- Abizena beti Zr zerrendan dagoela suposatzen da.
-- Zr zerrendan Abizena osagaiaren indizea itzultzen du.
begin
   Indizea:= Zr'First+Salto-1;
   while Zr'Last>Indizea and then Abizena>Zr(Indizea) loop
     Indizea:= Indizea+Salto;
   end loop;
   if Zr'Last>Indizea then Muga:= Indizea;
   else Muga:= Zr'Last; end if;
   return (BilaketaLineala(Zr(Indizea-Salto+1..Muga), Abizena));
end Bilatu;
```

a) Kasu txarrenean zenbat alderaketa egiten dira Abizena eta Z erreko osagaien artean? Ezaguna da *BilaketaLineala(Zr(X..Y),Abizena)*-ri eginiko deiak $Y-X$ alderaketa egiten dituela.

b) Algoritmoaren exekuzio ordena aldatuko litzateke *BilaketaLinealari* deia egin ordez *BilaketaDikotomikoa(Zr(X..Y),Abizena)*-ri egingo bagenio (jakinez honek $1+\lfloor \log (Y-X) \rfloor$ alderaketa egiten dituela)?

4) *n* giltza desberdinez osatutako sekuentzia bat izanik, ondoren ematen den *Txertaketa* ordenazio algoritmoaren aldaera azter bedi:

∀i 2≤i≤n: S(1)≤S(2)≤ ... ≤S(i-1) sekuentzian S(i) zuzenean bere posizio erlatiboan txertatzen da. BilaketaDikotomikoa erabiltzen da S(i)-ren posizio zuzena S(1)≤S(2)≤ ... ≤S(i-1) sekuentzian mugatzeko.

Hau da, Txertaketa algoritmoak *S(i)* osagaia zein posiziotan txertatu behar duen mugatzeko ondorengo algoritmo hau erabiltzen du:

```
proz PosizioaDikotomikoki(Sek, Balioa, Posizioa)
-- Balioa  sekuentzian ez dago
   N:=Sek'length; H:=Sek'First;
   A:=Sek'Last; E:= (H+A) div 2;
begin

  if N=1
  then  if    Sek(H) < Balioa   then  Posizioa:= H+1;
        else Posizioa:=H; end if;
  else
     if Sek(E)<Balioa
     then PosizioaDikotomikoki(Sek(E+1..A), Balioa, Posizioa);
     else PosizioaDikotomikoki(Sek(H..(E-1)), Balioa, Posizioa);
     end if;
  end if;
end;
```

Adibideak:

S

2	5	8	18	20	30	Balioa	…	S(n)
1	2	3	4	5	6	i	…	n

PosizioaDikotomikoki(S(1..6), 1, P) → P = 1

PosizioaDikotomikoki(S(1..6), 3, P) → P = 2

PosizioaDikotomikoki(S(1..6), 9, P) → P = 4

Ondorengoa aztertzea eskatzen da:

a) Giltzen arteko alderaketak: Zein da kasu txarrena? Kasu txarren horren exekuzio denboraren ordena zein da?

b) Giltzen mugimenduak: Zein da kasu txarrena? Kasu txarrena horretan, zenbat giltzen mugimendu egiten dira guztira?

c) Zein da algoritmoaren ordena kasu txarrenean?

5) Anderrek Z zenbaki osoko positiboa pentsatu du. Z zenbakia zein den asmatzeko Anderri hari buruz itaun diezaiokezu bakarrik >, < eta = erlazioak erabiliz. Idatzi galdeketa sistematizatuko duen algoritmo bat. Idatzitako soluzioak Anderri o(Z) galdera egin behar dizkio. Frogatu hala dela.

6) Bilaketa Dikotomikoaren estrategia erabiliz, emaniko N zenbaki arrunta ondorengo hiru zenbaki arrunten biderkatze den ala ez mugatuko duen algoritmoa idatzi, soluzioaren denbora-ordena o(N) dela ziurtatuz. Soluzioaren denbora-ordena kalkulatu.

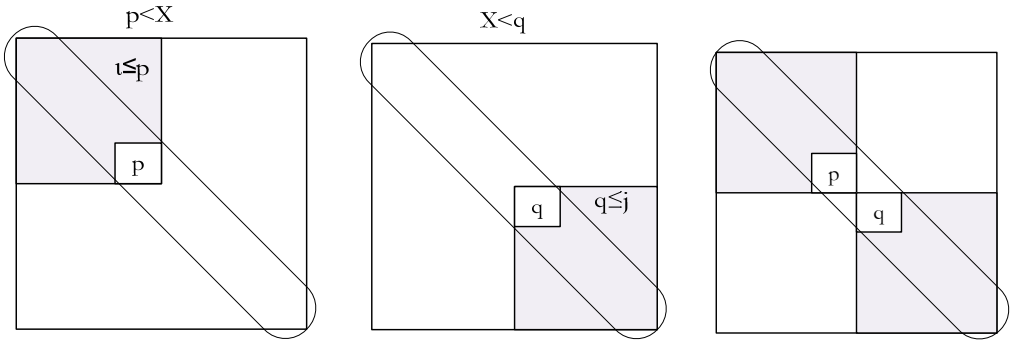
7) Demagun *V(1..N)* taulako zenbakiak txikitik handira ordenaturiko daudela bertako zenbait zenbaki negatiboak izan litezkeelarik. $i \in \{1,\ldots,n\}$ izanik, V(i)=i gertatzen den ala ez eraginkorki erabakiko duen algoritmoa egizu, dagoenean *i* itzuliaz. Soluzioa *o(n)* izan behar du (lineala baino hobea). Baldintza hori betetzen duen posiziorik existituko ez balitz, algoritmoak 0 itzuli beharko du.

8) Izan bedi `M(1..n, 1..n)` errenkadetako eta zutabeetako osagaiak ordena gorakorra jarraituz ordenaturik dituen matrize bat (ezkerretik eskuinera eta goitik behera). X osagaia M matrize horretan dagoen edo ez mugatzeko honako algoritmoa jarraitzen da:

   • Bilaketa dikotomiko bidez X diagonalean bilatzen da.

   • Baldin X aurkitzen badugu, amaitzen dugu.

   • Bestela, (*p* eta *q* aurkitzen ditugu non $p<X<q$) X egon ez daitekeen bi matrize zatiak baztertzen ditugu eta errekurtsiboki prozesu bera gelditzen diren beste bi zatitan aplikatzen da. Zein da algoritmoaren ordena kasu txarrenean?



9) $a_1, a_2, ..., a_n$ osokoen sekuentzia taula batean metaturik dago. Gainera, taulako edozein *ai* posizio jarraietan metaturik dauden sekuentziako osagaien balioak gehienez unitate batean desberdinak direla ezaguna da. Sekuentzian osagai jakin bat, *X*, ote dagoen eraginkorki mugatuko duen algoritmoa idaztea eskatzen da.

10) Osokoen bektore bat emanik, bertako osagai handiena eta txikiena zeintzuk diren erabakitzen duen algoritmo bat ebatzi. Bistakoa da *2n-2* alderaketa egiten dituen soluzioa. Hobea bilatu; hots, lineala bai, baina kopuru hori baino osagai arteko alderaketa gutxiago egiten dituena, adibidez ($3/2n \pm a$) egiten dituena.

11) *MergeSort* algoritmoetan zenbat osagaien mugimendu egiten da kasu txarrenean? Eta zenbat osagai arteko alderaketa? *BubleSort*, *SelectionSort* eta *InsertionSort* algoritmoen emaitzak eta honenak alderatu eta ondorioak atera. Osagaien arteko alderaketa kopuruak zenbatzea soilik, algoritmo hauek egiten duten lanaren adierazgarri ona da? Algoritmo guztietan?

12) *Mergesorte*-k memoria espazio estra lineala behar duela eta, gastu hori ekiditeko Mergesort-en ondorengo bertsio berria asmatu dugu: Bertsio berriak behar duen denbora ordena kalkula bedi.

```
procedure MSTaulaGainean (V: in out Tautla; I,J: in Indizea) is
  K: Integer:= (I*J) div 2;
begin
  if TxikiaDa?(I,J)  then  SortSinplea(V,I,J);
  else
     MSTaulaGainean(V,I,K);
     MSTaulaGainean(V,K+1,J);
     MSTaulaGainean(V,I,K,J);
  end if;
end MSTaulaGainean
```

Bertako *MSTaulaGainean(V, I, X, J)* prozedurak V-ren *V(I ..J)* azpitaula ordenatzen du *V* en bata bestearen segidan dauden *V(I..X)* eta *V(X+1..J)* segmentu ordenatuak ematen zaizkionean parametro bezala.

```
procedure MSTaulaGainean (T: in out Taula;  I,X,J: in Indizea) is
  Ezk: Indizea:=I; Esk: Indizea:=X+1; Lag: TElem;
begin
  while Ezk \= Esk loop
     if T(Ezk)>T(Esk)
     then  Lag:= T(Ezk);  T(Ezk):= T(Esk);
           Ordenatuta_utziz_Lag_T(Esk..J)n_txertatu
     end if;
     Ezk:= Ezk+1;
  end loop;
end MSTaulaGainean;
```

13) Osokoen taula bat emanik, bertako *K* osagai txikiena bilatuko duen algoritmoa idatzi eta ordena azter ezazu; hots:

m *Bek-eko K. osagai txikiena da* $\Leftrightarrow$ $\mathcal{N}i\{1 \le i \le n \wedge Bek(i) < m\} < K$
$\mathcal{N}i\{1 \le i \le n \wedge Bek(i) \le m\} \ge K$

K=m/2 denean, ebazten den problemari medianaren hautaketa deritzo. Honen adibide bi:

81	18	17	1	50	36	-8	

-8	1	17	18	50	36	81	90

14) (a) Ezaguna den bilaketa dikotomikoa errekurtsiboki kalkulatzen du ondorengo algoritmoak: (X bektorean ez badago, orduan 0 indizea itzuliko du)

```
proz    BilaketaDiko(V(i..j), X, Ind)
  if  i< j  then
     k:= (i+j) div 2;
     if  V(k)=X  then Ind:=k;
     else
       if  V(k)<X  then BilaketaDiko(V(k+1..j),X,Ind);
       else  BilaketaDiko(V(i..k-1), X, Ind);
       end if;
     end if;
  else
     if  i=j and then V(i)=X  then Ind:= i;
     else Ind:= 0;
     end if;
  end if;
end proz;
```

Algoritmoaren bi inplementazio desberdin kontsidera bitez, batean parametroen pasatzea erreferentzia bidez egiten dena eta bestean aldiz balio (edo kopia) bidez. Bi inplementazioen denbora-eraginkortasunean diferentziarik badago?

(b) Ohiko *MergeSort*-en azaldutako bi parametroen pasatze aukeren artean denbora-eraginkortasun diferentziarik badago?

15) 15 osoko dituen taula bat asma ezazu non taula hori *QuickSort*-i emanez gero *QuickSort*-en kasu onena den. Irudi bidez froga ezazu zure taula kasu onena dela.

16) Demagun bateraketa bidezko (*Merge*) ordenazio algoritmo bat erabili nahi dugula, baina bektorea hiru zatitan banatuz, errekurtsioz zatiak ordenatuz, eta ondoren hauek bateratuz.

a) `MergeSort3` algoritmoa idatzi eta aztertu.

b) Orain algoritmoa idatzi gabe, orokor ezazu aurreko prozesuaren analisia (kostua) 3 zatitan banatu ordez, jatorrizko bektorea K zatitan banatuko lukeena, eta hauek ordenatu ostean, K azpibektore ordenatuak bateratuko lituzkeena.

c) Aurreko analisitik zer ondorioztatzen duzu?

d) Eraman dezagun bektorearen zatiketaren ideia hori limiteraino, ondoren bateraketa bidez ordenatzeko. Horretarako `MergeSortApurrak` algoritmoa asmatzen dugu, non jatorrizko bektorea 2 osagai dituzten (n/2) zatitan banatzen duena, eta zati bakoitza ordenatu ostean, n/2 zatitxoak bateratzen dituena. Algoritmoa idatzi gabe, haren kostu funtzioa idatzi eta denbora ordena kalkulatu.

17) Aurreko ariketa kontuan izanik, `MergeSort`$\sqrt{n}$ algoritmoa asmatu dugu oraingoan: jatorrizko bektorea $\sqrt{n}$ tamaina duten $\sqrt{n}$ zatitan banatzen duena, eta $\sqrt{n}$ zatiak errekurtsioz ordenatu ostean bateraketa bidez bektore osoaren ordenazioa lortzen duena.

Ondorengo errekurtsioa gara ezazu:

T(n)= $\sqrt{n}$ * T($\sqrt{n}$) + n baldin n>1 T(1) = 1

Ebatzi duzun errekurtsio ekuazio, `MergeSort`$\sqrt{n}$ algoritmoari dagokio? Arrazoitu erantzuna.

18) *Mergesort* algoritmoaren *TA(n)*, *TM(n)*, *MEE(n)* eta ordena zehatzak (bai denborarako bai eta espazio estrarako) honakoak dira:

```
TA(n)= Kasu txarrenean alderaketa kopurua = (n-1) + 2T(n/2)
TM(n)= K. txarrenean mugimenduen kopurua = 2n + 2T(n/2)
```

Algoritmoak batez ere, eragiketa horiek egiten dituenez, algoritmoak egiten duen lana bi horien batuketa da: $T(n) = TA(n)+TM(n) \in \Theta(max(TA(n), TM(n)) = \Theta(n \lg n)$

$MEE(n) = n+a + MEE(n/2) \in \Theta(n)$

Orain, bertsio *MergeSort*-en aldaera berriak idaztea eskatzen da. *Merge* metodo berriak behar izanez gero hauek "idatzi". Ahalik eta onenak izan behar dute metodoek.

a) *Mergesort3*: tamaina berdineko *3* zatitan banatzen du sarrerako taula

b) *Mergesort4*: tamaina berdineko *4* zatitan banatzen du sarrerako taula

c) *Mergesort*$\sqrt{n}$: tamaina berdineko $\sqrt{n}$ zatitan banatzen du sarrerako taula

d) *Mergesort(n/2)*: tamaina berdineko *(n/2)* zatitan banatzen du sarrerako taula

e) *MergesortK-Heap*: tamaina berdineko K zatitan banatzen du sarrerako taula eta Meta egitura nonbait erabiltzen da. Bestera: Izan bitez K taula ordenatuak, guztira n osagai dituztenak. K osagaien arteko txikiena minimoen metak erabiliz muga dezakegu. Taula horien *merge* bidezko bateraketa metak erabiliz aztertu.

19) $1 \leq m \leq n$ izanik, n zenbaki osoko dituen bektore bateko m zenbaki txikienak mugatu nahi ditugu (adibidez, bektoreko aurrealdean kokatuz). Ez da beharrezkoa m osagaiak ordenaturik ematea.

   a) QuickSort algoritmoan inspiratuz, eta konkretuki, honen Banaketa prozedura erabiliz, problema hori ebazteko algoritmoa idatzi. Guztiz garrantzitsua da soberazko ordenazioak ez egitea.

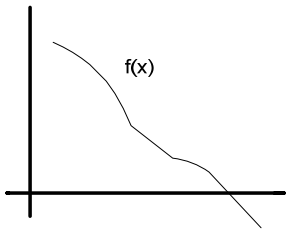
   b) Zure algoritmoaren exekuzio-ordena kasu txarrenean kalkula ezazu.

   c) Demagun zenbakizko bektore baten mediana kalkulatzeko Mediana algoritmo lineala dugula. Demagun prozedura hori lehenengo ataleko algoritmoan banaketa-puntua (pibotea) aukeratzeko erabiltzen dugula. Era horretan, bektorean pibotea baino txiki- agoak eta handiagoak diren osagaien kopuruak berdinak dira. Algoritmoa idatzi gabe, azter ezazu honen denbora ordena kasu txarrenean.

20) Argirik ez dagoen gela batean bi kaxoi ditugu: batean n torloju daude eta bestean aurrekoen n azkoinak. Problema torlojuen eta azkoinen doikuntza egitea da. Erlazioa biyektiboa da; hots, torloju bakoitzari azkoin bakarra dagokio eta alderantziz. Ilunpean gaudenez, aukera bakarra torloju eta azkoin parea hartzea eta estutuz probatzea da ea doitzen diren, azkoina handiegia den ala txikiegia den. Ataza burutzeko diseina ezazu algoritmo eraginkorra (bataz besteko kasuan $o(n^2)$ multzokoa den soluzioa).

21) $f(x)$ funtzio monotono beherakorra izanik, izan bedi n osokoa $f(n) \geq 0$ betetzen duen zenbaki osoko handiena. n existitzen duela asumituz, hura kalkulatzen duen algoritmoa honako hau da:

```
I:=0;
while F(I)>=0 loop
   I:= I + 1;
end loop;
N = I-1;
```



Halere, soluzio horren ordena O(n) da. Portaera asintotiko hori (n-ren funtziopean) hobetzen duen algoritmoa idaztea eskatzen da.

22) Osokoen taula bat emanik, batura maximoa ematen duen segmentua mugatu nahi da. Balioa eta segmentua kalkulatuko dituen metodoa idatzi eta aztertu. Adibidea: *baturaMax=29* litzateke honako sekuentziarentzat

22	37	8	12	1	-10	18	-25	20	-3

23) Osokoen $A=[a_1,a_2,\ldots,a_n]$ bektorea emanik, A-n dauden osoko berdinen segmentu luzeena mugatzeko algoritmo bat diseina ezazu *zatitu eta irabazi* teknika erabiliz. Algoritmoa optimoa dela uste duzu? Zure erantzuna arrazoitu.

24) Ondorengo algoritmoak sarrerako hitza palindromoa den mugatzen du:

```
function Pal (H:    Hitza; i,j: Indizea) return boolean is
begin
  if i≥j then return true;
  elsif H(i)≠H(j) then return false;
  else return Pal(H, i+1, j-1);
  end if;
end;
```

Aztertu P al(H it, 1, n) deiak behar duen denbora eta denbora-ordena kasu txarrenean eta batez besteko kasuan. Suposatu sarrera guztiak ekiprobableak direla eta kateen alfabetoa {a, b} dela.

25) Izan bedi $T(1..N)$ zenbakien bektore bat. $Kard(i|T(i)=X) > n/2$ propietatea betetzen duen X osagaia mugatuko duen algoritmoa idaztea eskatzen da; hau da, n/2 aldiz edota maizago agertzen den X osagaia T-n. Halere, T-ko N osagaien artean propietate hori betetzen duen X osagairik ez egotea gerta liteke.

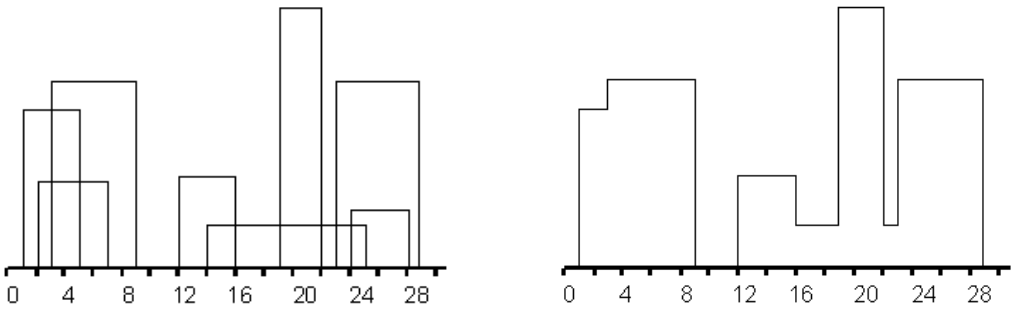
26) $A(a-1..b+1)$ taulako minimo lokala $A(k-1)\geq A(k)\leq A(k+1)$ baldintzak betetzen dituen $A(k)$ osagaia da. $a\leq b$, $A(a-1)\geq A(a)$ eta $A(b)\leq A(b+1)$ betetzen direla suposatuko dugu; baldintza hauek behintzat minimo lokal bat existitzen duela ziurtatzen dute. $A(a-1..b+1)$ tarteko minimo lokal bat mugatuko duen algoritmo bat idaztea eskatzen da.Soluzioa nabariak kasu txarrenean O(n) denbora hartzen duenez, zure soluzioa hori baino hobea izan beharko du. Horrela dela arrazoitu.

27) Ordenaturik dagoen V(1..n) bektorea emanik eta Z eta S bi balio (non $Z\leq S$), Z eta S balioen arteko balioz osaturiko V-ko azpibektorearen bi indizeak mugatzeko algoritmo bat egizu bilaketa dikotomikoaren ideia erabiliz. Soluzioak kasu txarrenean O(lg n) ordena duela frogatu. Adibidez, sarrera V=(3, 3, 3, 5, 5, 5, 7, 8, 8, 10, 12, 12, 12, 12, 12); Z=5 eta S=8 denean, irteerak (4,9) izan behar du.

28) N partaideko liga bidezko txapelketa bat antolatu behar dugu. Lehiakide bakoitzak eguneko partidu bakarra jokatuko du (salbuespen bakarra egongo da n bakoitia denean, egun bateko atsedena izango du). Lehien egutegi bat prestatu behar dugu, non denak denen kontra jokatu behar duten eta (**n-1**) egunetan amaitu beharko duena (egun bat

gehiago *n* bakoitia bada). *N* partaideak emanik eta zatitu eta irabazi teknika erabiliz, lehien egutegi bat ekoitziko duen algoritmoa egizu.

29) *Skyline* (poligonoen batura). Etxe-orratzak **(x,h,y)** hirukote bidez adieraz litezke; hau da, urrutitik begiratuta abszisa-ardatzarekiko **h** etxearen altuera litzateke eta **x** eta **y** balioak etxe-orratzaren hasiera eta amaiera puntuak lirateke ardatza horretan. Demagun $\mathbf{(x_i,h_i,y_i)}$ hirukoteen sekuentziak ($1 \le i \le n$) hiri bateko **n** etxe-orratz jasotzen dituela. Suposa dezakezu **n** etxe-orratzen sekuentzia ordenatuta dagoela etxe-orratzen hasierarekiko (x_i balioekiko). Helburua hiriko etxe-orratzek marrazten duten zeruertz-lerroa kalkulatzen duen algoritmoa egitea da; hots, ezkutuan gelditzen diren marrak ezabatuz zeruko perfila bi dimentsiotan kalkulatzen duen algoritmoa. Zeruertz-lerroaren adierazpideak (z_1, k_1, z_2, k_2,…, z_{m-1}, k_{m-1}, z_m) itxura izan dezake. Honek adieraziko luke, zeruertz-lerroa z_1 abszisan hasten dela k_1 altueraraino igoz, hau z_2 abszisaraino mantentzen dela, eta puntu horretan altuera k_2 izatera aldatzen dela, eta horrela behin eta berriro, azkeneko eraikina z_m abszisan amaituz. Adibidea: Adibidea, ((1,**11**,5), (2,**6**,7), (3,**13**,9), (12,**7**,16), (14,**3**,25), (19,**18**,22), (23,**13**,29), (24,**4**,28)) etxe-orratzei dagokien zeruertz-lerroa litzateke: (1,**11**,3,**13**,9,**0**,12,**7**,16,**3**,19,**18**,22,**3**,23,**13**,29,**0**), eta grafikoki:

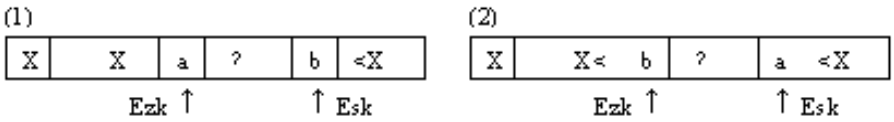


30) Problema hau *QuickSort*-ko Banatu prozeduraren aukera berri bat da.

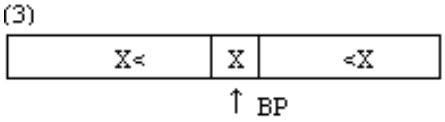
```
procedure QuickSort (Bek: in Sekuentzia) is
  BP, H,B: SekIndizea;
begin
  H:=Bek'First; B:=Bek'Last;
  if  H<B  then     Banatu2 (H,B,BP);
                    QuickSort(Bek(H, BP-1));
                    QuickSort(Bek(BP+1, B));
  end if;
end QuickSort;
```

Prozesuaren bertsio berria: *X=Bek(H). X*en balioaren funtzioan Bek sekuentziako balioak bi azpisekuentziatan banatzeko modu egoki bat, *Bek(H, B)* behin bakarrik korritzea da, baina sekuentziaren bi ataletatik hasita. Bi indize-aldagai behar dira, Ezk eta Esk. Indize hauek *H* eta *B+1* balioekin hasieratzen dira hurrenez hurren. Ezk inkrementatu egin behar da *Bek(Ezk)>X* gertatu arte eta, aldiz, Esk dekrementatu

Bek(Esk)≤*X* gertatu arte (1 irudia ikusi). Orduan *Bek(Ezk)* eta *Bek(Esk)* elkarraldatzen dira (2 irudia). Prozesuak *Ezk*<*Esk* baliozkoa den bitartean jarraitzen du.



Azkenean, Bek(H) eta Bek(Esk) permutatzen dira X bere posizio zuzen erlatiboan kokatzeko, hots, BP posizioan. (3 irudia)



31) Prozesuaren kodea ondorengo prozeduran jaso da:

```
procedure Banatu2 (Hasi,Buka: in SekIndizea;BP: outSekIndizea) is
  Ezk: SekIndizea:= Hasi;  Esk: SekIndizea:= Buka+1;
  X: SekBalioMota:= S(Hasi);

begin
  loop
     Ezk:=Ezk+1;
     exit when S(Ezk)>X or Ezk≥Buka;
  end loop;
  loop
     Esk:= Esk -1;
     exit when S(Esk)<=X;
  end loop;

  while  Ezk<Esk loop
     BekTaualanElkarraldaketa(Ezk, Esk);
     loop
        Ezk:=Ezk+1;
        exit when S(Ezk)>X
     end loop;
     loop
        Esk:= Esk-1;
        exit when S(Esk)≤X;
     end loop;
  end loop;
  BekTaualanElkarraldaketa(Hasi, Esk); BP:=Esk;
end Banatu2;
```

Galderak: n osagai dituen sekuentzia bat emanik:

a) *Banatu2* algoritmoan eta kasurik txarrenean, *Bek*-eko osagaien artean zenbat elkarraldatze egiten dira?

b) *QuickSort* algoritmoan eta kasurik txarrenean, *Bek*-eko osagaien arteko elkarraldatze kopuruaren ordena zein da?

c) *Bek*-ko osagaien arteko zenbat alderaketa egiten ditu kasu tzarrenean *Banatu2*-k? Hots, *Banatu2(BektF irst, BektLast, BanaketaP)* deiak *Bek*-eko osagaien artean egiten dituen alderaketa kopurua kasu txarrenean.

d) *QuickSort* algoritmoan eta kasurik txarrenean, *Bek*-eko osagaien arteko alderaketa kopuruaren ordena zein da?

e) *QuickSort*-ek sarrerako sekuentzia banatzeko goran azaldu den bertsioa erabiliko balu, bere ordena kasu txarrenean aldatuko litzateke? Eta batez besteko kasuan? Gehienez 10 lerrotan erantzuna arrazoi bedi.

32) Bektore bateko mediana, bektorea ordenatuta balego $\lfloor n/2 \rfloor$ posizioan legokeen balioa da. Honela, (2,4,6,5,3,1) bektoreko mediana 3 da.

a) Emaniko bektorearen mediana kalkulatuko duen algoritmoa idatz ezazu. Algoritmoaren konplexutasuna aztertu eta kalkulatu.

b) Orain, zatitu eta irabazi teknika bidez eta BANAKETA prozedura erabiliz *mediana*-ren problema ebazten duen metodoa garatu eta bere denbora-kostua kalkula ezazu kasu onenean eta txarrenean.

c) Demagun Θ(n) kostua duten bi prozedura ditugula dagoeneko definituak: MEDIANA (V(1..n),M), ematen zaion bektorearen mediana kalkulatzen duena M-en eta ANTOLATU(V(1..n),P) V bektoreko osagaien permutazio bat itzultzen duena V-en honako propietateak betetzen dituenak:

i) Baldin $V(i)<P$ eta $V(j)=P$ orduan $i<j$

ii) Baldin $V(i)=P$ eta $V(k)>P$ orduan $j<k$

d) MEDIANA eta ANTOLATU prozedurak erabiliz *QuickSort*(V(1..n))-en beste bertsio bat egizu kasu txarrenean o(n2) dena. Soluzioak aipatutako ordena betetzen duela frogatu behar duzu.

33) Aire kontrolerako sistema batek $\boldsymbol{n}$ espazio-ontzi ditu zirkulazioan bere ekintza-eremu barruan eta espazio-ontzi bakoitzaren $\boldsymbol{(x,y)}$ koordenatuak ezagunak izanik. Informazio hau guztia pantailaratu egiten da giza kontrolatzaile batek gertatzen denaren ikuspegi orokorra izan dezan. Dena dela, elkarren artean oso hurbil dauden hegazkinenei arreta berezia eman nahi diegu, elkar talka egiteko arrisku handiagoa baitute. Horretarako sistemak gorriz eta ñir-nir eginez elkarren artean hurbilen dagoen hegazkin parea adieraz zezan nahiko genuke. Egizu algoritmo bat kalkula dezan hegazkin pare hori, soluzioaren denbora-ordena kalkulatuz.

ALGORITMOEN DISEINUA
G5: PROGRAMAZIO DINAMIKOAREN TEKNIKA

1) *f(1)=1* izanik, zein programazio teknika erabiliko zenuke $f(n) = n + \frac{2'}{n} \sum_{i=1}^{n-1} f(i)$ kalkulatuko lukeen programa bat idazteko? Programa horren exekuzio ordena zein izango litzatekeen laburki arrazoitu.

2) Demagun *Partiketa* funtzioa honela definituta dagoela:

```
type Bektore is array (Integer range <>) of Positive;
function Partiketa (B: Bektore ) return Integer is
begin
  for I in 1.. N loop
    if Batu(J=1,I-1,B(J))=Batu(J=I,N,B(J)) hen return I;
    end if;
  end loop;
  return 0;
end Partiketa;
```

Zenbaki osoko positiboen B bektorea emanik, Partiketa funtzioak ondorengo baldintza betetzen duen I (B bektoreko) posizioa itzultzen du:

I posizioaren ezkerraldean dauden B-ko osagaien batura (B(I) balioa kanpo) eta
I posizioaren eskuinaldean dauden B-ko osagaien (B(I) balioa barne)
baturak berdinak dira.

Aurreko baldintza betetzen duen posiziorik ez badago bektorean, funtzioak 0 itzuli beharko du.

a) Frogatu azaldutako ezaugarriak betetzen dituen posizio bakar bat existi daitekeela gehienez.

b) Programaren exekuzio-denbora eta denbora-ordena kalkula bitez.

c) O(*n*) ordena hartuko duen algoritmo bat idatz bedi

d) O(*lg n*) denbora hartuko lukeen algoritmorik aurkitzea posible ote litzatekeen laburki eztabaidatu.

3) $m \times n$ tamainako labirinto batean gora, behera, eskuinera eta ezkerrera mugimenduak bete ditzake errobota batek baldin eta noranzko horretan ormarik ez badago. Labirinto konkretu bat emanik, irteera posizioa eta helmuga posizioak emanik, zein agindu

sekuentziak bete behar dituen errobotak kalkulatuko duen algoritmoa idatzi backtrack eskema erabili gabe.

4) *X* eta *Y* zenbaki positiboak izanik, demagun ondorengo programa:

```
function Nagusia (Znbk: positive) return real is
  Taula: array (1..Znbk) of real:=(1=>X;2=>Y; others=>0);
  function LAG (Znbk: natural) return real is
  begin
    if Taula(Znbk-1)=0 then Taula(Znbk-1):= LAG(Znbk-1); end if;
    if Taula(Znbk-2)=0 then Taula(Znbk-2):= LAG(Znbk-2); end if;
    Taula(Znbk):= (Taula(Znbk-1)+Taula(Znbk-2))/2;
  end LAG;
begin
  if Znbk≤2 then return Taula(Znbk);
  else return LAG(Znbk); end if;
end Nagusia;
```

a) *Nagusia(n)*-k kalkulatzen duen funtzioa definitu.

b) Aurreko programaren diseinurako, zein teknika erabili da?

c) *Nagusia(n)*-ren egikaritzapen-denbora ordena kalkula ezazu.

d) Sarrerako datuak *X=1* eta *Y=1* balira, aurreko programa erabiliko zenuke? Zergatik?

5) (a) $b_1,b_2,...,b_N$ *N* txanponen klaseak emanik eta jakinik txanpona klase bakoitzeko behar hainbat txapona eskuragarri ditugula, *K* kopuru finkoa emango duten *txanpon kopuru* minimo bat kalkulatuko duen algoritmo bat idaztea eskatzen da. Algoritmoaren *T(N)* kalkulatu. (*K* konstantea dela suposatuko da.).

(Aholkua: izan bedi *Z(K) K* batzen duten txanpon zenbakirik txikiena. Aztertu bedi zer erlazioa dagoen *Z(K)* eta $Z(K-b_i)$-ren artean, non $1 \le i \le n$ eta $b_i=K$).

(b) Eta *i* klase bakoitzeko txanpona kopurua m_i balioaz mugaturik balego? Azter ezazu egoera berria eta muga horiek kontuan hartzen dituen ekuazio sistema ere defini ezazu.

6) (a) $b_1,b_2,...,b_N$ *N* txanponen klaseak emanik eta jakinik txanpona klase bakoitzeko behar hainbat txapona eskuragarri ditugula, *K* kopuru jakin bat itzultzeko zenbat txanpon *konbinazio* desberdin dauden kalkulatuko duen algoritmo bat idaztea eskatzen da.

Adb.: Txaponen klaseak honakoak balira: $b_1= 25$, $b_2= 50$, $b_3= 75$ eta $b_4= 100$

Balioa	Txanpon konbinazio desberdinak
100	5 modu: 25 + 75, 2 • 50, 100, 4 • 25, 50 + 2 • 25
116	0, ez dago konbinaziorik

Iradokizuna: 25+75 konbinazioa 75+25 konbinazioaren berdina da, eta ondorioz, konbinazio bakarra kontsideratu behar da.

(b) Eta *i* klase bakoitzeko txanpona kopurua m_i balioaz mugaturik balego? Azter ezazu egoera berria eta muga horiek kontuan hartzen dituen ekuazio sistema ere defini ezazu.

7) *N* objektuk p_1, p_2, ..., p*n* pisuak eta b_1, b_2, ..., b*n* balioak erlazionaturik dituzte hurrenez hurren. Bestalde, *E* edukiera duen motxila bat ere badugu. Helburua, motxilaren edukiera gaindituko ez duten objektuen azpimultzo bat bilatzea da haien balioen batura maximizatuz. (Backtracking eskema erabiltzen dituzten soluzioak ez ditugu onartuko.)

8) $\{b_1,\ldots,b_n\}$ balioak dituzten *n* zigilu mota ditugu. *Z* posta zenbatekoa duen eskutitza, zenbat modu desberdinetan frankea dezakegu? (Postaren frankeoa zehazki *Z* izan behar du, eta noski, zigiluen itsaspen ordenak ez du garrantzirik).

   a) Programazio dinamikoa teknika erabiliz gara ezazu soluzio bat eta haren denbora ordena eta memoria espazio estra kalkula itzazu.

   b) Gainera, defini dituzun indukzio ekuazioak erabiliz eta honako datuokin (b_1=2, b_2=3, b_3=5 eta b_4=8 zigilu motak eta *Z*=21 posta zenbatekoa) dagozkion balioz osatu memoria metatze egitura (iteratiboki edo errekurtsiboki).

9) Unibertsitate bateko sail batek *IK* ikastordu eman behar ditu. Horretarako, *N* klaseko desberdinetako irakasleak kontrata ditzake. *I* klaseko irakasle bakoitzak H_i ordu irakatsi ditzake gehienez lan horregatik beti P_i ogerleko eskuratuz. *IK* klaseak emateko eta ordaindu beharreko ogerleko kopurua minimoa izan dadin, zenbat irakasle kontratatu beharko liratekeen kalkulatuko duen algoritmo bat idatzi. Aldi berean, *N* irakasle klase existitzen direla jakinda, algoritmoaren exekuzio-denboraren ordena kalkulatu.

   Aurreko algoritmoa egoki ezazu (eta berridatzi), gainera, klase bakoitzeko zenbat irakasle kontratatu behar diren jakiteko.

10) Lagunak lanean enpresako zuzendariak Marigorriri aholkularitza bat egin dezala eskatu diola eta, honi, enpresa afari bat antolatzea bururatu zaio. Gonbidatuen zerrenda egiteko, gainbegirale funtzio bat erabil dezake. Enpresako langileriaren egitura hierarkikoa denez, funtzio honen heina piramide bat osatzen du haren gailurrean gaur egungo zuzendaria egonik. Bestalde, administrazioko langileek, langile guztiek erlazionaturik duten jendetasun balioarekiko, lantegiko langile guztiak ordenatu dituzte zerrenda bat osatuz. (Jendetasun balioa zenbaki erreala da.) Afarira joango diren partaideak, ekintza hura goza dezaten, zuzendariak langile bat eta honen gainbegirale zuzena, biak, afarian egon ez daitezela murriztapena ezarri du:

   a) Zerrenda kalkulatuko duen algoritmoa idatz bedi. Helburua, partaideen jendetasun balioa maximizatzea izango da. Exekuzio denbora orden kalkula bedi.

b) Marigorrik nola ziurta dezake partaideen artean enpresako zuzendaria egongo dela?

11) *K* hasierako kopuru bat emanik eta honi *×2, ×3, ×5* eragiketak behin eta berriro aplikatuz, zer balio *X* lor dezakegu *M* baliotik hurbilen dagoena baina hau gainditzen ez duena? Programazio dinamikoa teknika erabil ezazu *X*-ren balioa mugatzeko eta proposatutako algoritmoaren denbora-ordena kalkulatu.

Aurreko algoritmoa egoki ezazu beharrezkoak diren aginduak gehituz, gainera, Z balio maximo hori ematen duten biderkagaiak zeintzuk diren jakiteko.

12) Demagun *K* ogerleko ditugula eta *n* produktu eskaintzen dizkigutela non p_1,..., p_n hauen prezioak diren. Programazio dinamikoaren teknika jarraituz diseinatu algoritmo bat azaldutako *K* kopuruarekin zenbat produktuen *erosketa konbinazio desberdin* egin ditzakegun mugatzeko.

Adibidez:

K=15 eta [p_1= 10, p_2= 7, p_3= 3, p_4=5] kopurua eta produktuak edukiz hurrenez hurren, ondorengoek 3 erosketa desberdin adierazten dituzte:

```
1) [p1= False, p2= False, p3= False, p4= False]
2) [p1= False, p2= True, p3= True, p4= True]
3) [p1= True, p2= False, p3= False, p4= False]
```

Zure algoritmoak zer erantzungo luke adibide honentzat.

13) Izan bedi *T N* zenbaki osoko desberdinez osatutako taula bat. *T*-ko osagaiak (eta bertan agertzen diren ordena mantenduz) ordena gorakorra jarraituz ordenaturik dauden sekuentzia desberdinen artetik sekuentzia luzeena kalkulatzea eskatzen da. Adibidez, sekuentzia 11,17,5,8,6,4,7,12,3 denean, algoritmoak 5,6,7,12 sekuentzia itzuliko du. Algoritmoaren denbora-ordena zenbatekoa da?

14) L_1, ..., L_n luzera osokoak dituzten *n* barra ditugu. *K* luzerako barra bat behar dugu. Barrak ezin dira moztu baina bai soldatu haien ertzetatik. *K* luzera duen barra eraikitzeko gure barren azpimultzorik existitzen duen ala ez mugatuko duen soluzioa garatu programazio dinamikoaren teknika erabiliz. Algoritmoaren exekuzio ordena eta erabili- tako memoria espazio estra.

OHARRA: Guztiz garrantzitsua da programazio dinamikoaren teknikaren pausoz pausoko bilakaera zuzena egitea.

15) S-ko azpisekuentzia bat S-ko edozein posizioetatik edozein osagai kopuru kenduz lortzen da. Adib.: *bbacabxxb* sekuentziako azpisekuentzia bat *acxb* da. Bi sekuentzia

emanik hauen azpisekuentzia komun luzeena mugatzen du ondorengo f funtzio errekurtsiboaren definizioak (kasuka):

f(R,ε)=ε

f(ε,S)=ε

f(a•R,a•S)= a•f(R,S)

f(a•R,b•S)=f(R,b•S) eta f(a•R,S) azpisekuentzien artetik luzeena *(a = b)*

Adib.: *f(belarri, berdela) = bela*, *f(kamamila, motxila) = mila*, *f(bai, ez) = ε*.

(Iradokizuna: Luzera luzeeneko azpisekuentzia bat baino gehiago existitzen badu, azpisekuentzia bakarra kalkulatzen du.)

f(R,S) sekuentziaren *Luzera* kalkulatuko duen eta programazio dinamikoa teknika jarraituz diseinaturiko algoritmoa idaztea eskatzen da (memoriadun funtzioak erabili gabe). Zein da algoritmoaren exekuzio-denbora ordena? Eta erabilitako memoria espazio estraren ordena?

16) Izan bedi *G* grafo zuzendua: *G=(Erp, Ark*). *G* grafoaren adierazpena Auzokideen ma trize baten bidez ematen da. Problema *Dij* matrize logikoa definitzea da, non *Dij* egiazkoa izango den *i*-tik *j*-ra gutxienez bideren bat existitzen bada eta faltsua bestela. Egoki bedi *Floyd*-en algoritmoa problema hau ebatz dezan. Bere exekuzio-denbora azter bedi.

17) *G=(Erp, Ark)* grafo zuzenduaren arkuen pisuak negatiboak ez direla dakigu eta $|Erp| = n$. G grafoa L auzokidetza matrize bidez emanik dator. i, j ∈ Erp multzoko erpin bikote bakoitzaren arteko distantzia minimoa kalkulatu nahi dugu honako formula erabiliz:

D_{ij}^{p}: *i*-tik *j*-rainoko distantzia gehienez *p* pauso eman ditzakegunean.

Oinarrizko balioak $D_{ij}^{p}=Lij$ dira eta *i* eta *j* guztientzat D_{ij}^{n-1}balioak bilatu nahi ditugu, jakinez D_{ij}^{p} errekurtsiboki honela definitzen dela.

Problema programazio dinamikoa teknika erabiliz ebaztea eskatzen da (memoriadun funtzioak erabili gabe), non goian azaldutako ekuazio errekurtsiboa jaso beharko duen; hots, ondorengo puntu guztiak eskatzen dira beraz:

a) tarteko emaitza zein datu-egituratan metatuko den mugatzea: bektorea ala matrizea behar den, egituraren posizio bakoitzak zer adierazten edo metatzen duen, egituraren haizeratzea, nola betetzen joango den, soluzioa non dagoen.

b) emandako definizio errekurtsiboa jarraituz (a)-n aukeratutako datu-egitura betetzen duela ikusi eta soluzioa itzultzen duen algoritmo iteratiboa eman.

c) algoritmoaren denbora eta espazio estra ordenak kalkulatu: O(T(*n*)) eta O(MEE(*n*)).

18) Nilo ibaian *N* kai daude, bakoitzean Y ontzi aloka daitezkeelarik ibaian behera dauden beste kaietara joateko (ibaian gora ezin da joan, korronte handia du eta ibaiak). Edozein A abiapuntu-kaitik ibaian behera dagoen edozein H kaira iristeko ontzi bakar bat alokatzea garestiagoa gerta liteke Atik A+1ra, A+1etik A+6ra eta A+ 6tik H-ra joatea baino, adibidez.

A abiapuntu-kai guztietatik H helmuga-kai guztietara joateko bidai kostu posible guztien artetik bidai kostu minimoak kalkulatzea eskatzen da, bai funtzio memoriadunak erabiliz, bai hauek erabili gabe, baina bi kasuetan programazio dinamikoa teknika erabiliz. Ondoren, exekuzio-denboraren eta memoria-espazio estraren ordenak kalkula bitez.

19) Arkuen pisuak positiboak dituen grafo zuzendu azikliko bat emanik, *Floyd*-en algoritmoak erpin bikote guztientzat bikoteko erpinen arteko distantzia minimoak kalkulatzen ditu. *Floyd*-en algoritmoa eguneratzea eskatzen da, egiten duenaz gain, distantzia minimoa duten bide kopurua kalkula dezan.

20) *N* biteko eta jarraian bi *1*eko bitak ez dituzten kate guztiak eraiki nahi ditugu. *Programazio dinamikoa*ren teknika erabil ezazu kalkulatzeko eraiki genitzakeen horrelako kate desberdinen kopurua baina kateak eraiki gabe. Algoritmoaren denbora-ordena eta erabilitako memoria espazio estra kalkula itzazu.

21) A alfabetoa A={a,b,c} izanik, ondorengo taula, alfabetoko bi karaktereen konbinazioak zertara berridatz daitezkeen jasotzen du:

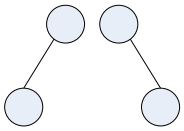
aa »» b	ab »» b	ac »» a
ba »» c	bb »» b	bc »» a
ca »» a	cb »» c	cc »» c

Berridazketa eragiketak, nahiz alfabetoan itxian izan, ez ditu trukaketa eta elkarketa propietateak betetzen.

$X=x_1...x_n$ karakterez osaturiko karaktere katean parentesiak zein posiziotan idatzi beharko liratekeen katea a-ra berridazteko mugatuko du idatzi behar den algoritmoak. Horrela, X= *bbbba* bada, algoritmoak bai itzuli beharko luke eta, adibidez, *(b(bb))(ba)* »» a (berridazketa ordena bat baino gehiago existi baitaiteke: *(b(b(b(ba)))) »» a*. *N* berridatzi behar den karaktere katearen luzera bada, algoritmoaren denbora-ordena zein da?

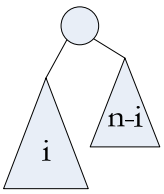
22) $A_1,A_2,...,A_n$ matrize segida biderkatzean, biderkatze eskalar kopurua minimizatzen duen algoritmoa gogora ezazu. Algoritmo horrek *Faktor(1..n, 1..n)* taula eraikitzen du, non *Faktor(i, j) = k* baldin *i*-garren eta *j*-garrenaren artean dauden matrizeen faktorketa optimoa $(A_i...A_k)(A_{k+1}...A_j)$ bada. *Faktor(1..n,1..n)* taula emanik $A_1,A_2,...,A_n$ biderkaduraren parentesiak jartzeko era optimoa inprimatzen duen algoritmoa idatz eta azter ezazu.

23) **N** adabegi dituzten zenbait zuhaitz bitar desberdin osa daitezke? Izan bedi Z(n) kopuru hori. Badakigu Z(0)=1 dela -zuhaitz hutsa-, Z(1)=1 -erroa soilik duen zuhaitza-, eta Z(2)=2 dela:



Z(3) eta Z(4) kalkulatzeko zuhaitzak marraz ditzakezu:

a) Z(n) kalkulatzen duen definizio errekurtsiboa idatzi ondorengo diagrama kontuan edukiz:



Orain errekurtsio-ekuazioa zuzena dela n=3, 4 balioentzat ziurta dezakezu zure marrazkiekin.

b) Programazio dinamikoa teknika erabiliz eta zuk aurreko atalean definitutako errekurtsioa jarraituz Z(n) kalkulatzen duen algoritmoa idatzi. Soluzioaren konplexutasuna kalkulatu.

24) Ariketa honetarako *Bikotetar* zuhaitzak erabiliko ditugu, non zuhaitz bat *Bikotetarra* den baldin eta zuhaitzeko barne adabegi orok zehazki bi ume baditu. Hori horrela, N hosto dituzten zenbat *Bikotetar* zuhaitz eratu litezke?

Programazio dinamikoaren teknika erabiliz zenbaki hori kalkulatuko duen algoritmoa egizu. Algoritmoaren denbora-ordena eta erabilitako memoria espazio estra kalkula itzazu.

25) $[b_i,b_i)$ $1 \le i \le n$ ekintzak izanik, ekintza-hautaketa problema programazio dinamikoa teknika bidez ebaztea eskatzen da. Algoritmoa $Zenb_i$ i=1,2,..,n zenbakiaren kalkulu iteratiboan oinarrituko da, non $Zenb_i$ zenbaki horrek $1,2,..,i$ ekintzen multzoko kardinalitate handieneko eta elkar bateragarri diren ekintzen kopurua metatuko duen. Ekintzak bukaera denbora balioekiko ordena gorakorra jarraituz ordenatuta daudela suposatu: $b_1 \le b_2 \le ... \le b_n$.

Soluzio honen eta problema beraren teknika jale bidezko soluzioaren (hain zuzen, lehenen amaitzen den ekintza eta aldez aurretik onartutako ekintzekin teilakatzen ez den hautaketa funtzioa duen soluzio jalea) exekuzio-denbora eta exekuzio-ordena konpara bitez.

26) Al'Telmoko zinemaldi maratoia hurbil dagoela eta, honek irauten dituen hiru egunetan zehar zeintzuk filma ikusiko ditugun planifikatu behar dugu kontu handiz, aldez aurretik sarrerak erosi ahal izateko. i pelikula bakoitza ($1 \le i \le n$) behin soilik proiektatuko da, HT_i unean hasiko delarik eta BT_i bukatuz. Gure iritzi-emaile gogokoenak pelikula bakoitzari Si izar esleitu dizkio, eta gure helburua ikusiko ditugun pelikulen izarren kopurua maximizatzea da.

Programazio dinamikoaren teknika aplika ezazu aipatutako helburua betetzen duen programazio bat egiteko, jakinik bi pelikulen artean ordubeteko tartea utzi behar dugula atsedena hartzeko.

Sinplifikatzeko ordu guztiak zinemaldia hasten den eguneko 0:00 orduarekiko neurtuko dira; hots, 0tik 72ra. Gainera, orduak osoak edo erdiak izango dira soilik, hau da, *i* pelikularen hasiera ordua HT_i=34,5 balitz, bigarren eguneko goizeko 10:30etan hasiko dela adieraziko luke.

27) Lantegi bateko gerenteak egun konkretu batean zeintzuk produktu laboratzea komeni den planifikatu behar du. Enpresako datu-basean honako informazioa dago:

- *T[1..n]* taula, non *T[i]*-k *i* klaseko ($1 \le i \le n$) produktu bat laboratzeko behar diren minutuak jasotzen duen,
- *On[1..n]* taula, non *On[i]*-k *i* klaseko ($1 \le i \le n$) produktu baten fabrikazioak ematen duen onura jasotzen duen, eta
- *Os[1..n]* taula, non *Os[i]*-k egun horretan *i* klaseko ($1 \le i \le n$) gehienez labora litezkeen produktu kopurua adierazten duen, lantegiko oinarrizko osagaien izakinak kontuan izanik.

Egun horretan eta planifikazio egoki baten ondorioz lantegiak lor dezakeen onura maximoa kalkulatuko duen algoritmo bat diseina ezazu programazio dinamikoaren teknika erabiliz.

Onura maximoaz gain, aurreko algoritmoa egoki ezazu kalkula dezan zeintzuk produktu komeni diren fabrikatzea onura hori lortzeko bai eta klase bakoitzeko zenbat produktu ere.

28) Garraio enpresa batek *M* kaxaz osatutako bidalketa garrantzitsu bat egin behar du. Bidalketa egiteko *N* garraio gailu mota du. i garraio motak K_i kaxa garraia ditzake gehienez. i garraio gailu baten desplazamenduaren kostua finkoa da, nahiz ia hutsik nahiz topetaraino beterik joan, P_i izanik kostu hori. *M* kaxak kostu minimoz garraiatuko duen garraio konbinazioa kalkula ezazu algoritmo batez eta soluzioaren denbora eta memoria kostuak aztertu.

29) Ariketa honetarako Bikotetar zuhaitzak erabiliko ditugu, non zuhaitz bat Bikotetarra den baldin eta zuhaitzeko barne adabegi orok zehazki bi ume baditu. Hori horrela, N hosto dituzten zenbat Bikotetar zuhaitz eratu litezke?

Programazio dinamikoaren teknika erabiliz zenbaki hori kalkulatuko duen algoritmoa egizu. Algoritmoaren denbora-ordena eta erabilitako memoria espazio estra kalkula itzazu.

30) Inbertsio enpresa batek fondo bat du *M* hamarreko eurokoa *P* produktu finantzario artean inbertitzeko. Produktu bakoitzak errentagarritasun espezifikoa du hartan inbertitutako diruaren arabera aldakorra dena. Enpresak matrize bat eraiki du *Errenta*(1..*M*,1..*P*), non *Errenta*(*k*,*f*) urteko mozkinaren portzentaia den, *k* hamarreko euroko ezarpena egiten denean *f* produktu finantzarioan. (Adibidez, *Errenta*(70,3)=4 bidez adierazten da, 700 euro 3 produktuan inbertituz, %4 mozkina espero denez, 28 euroko onura lortuko dugula.) Oharra: garrantzitsua da jabetzea *f* produktu bakoitzeko mozkinaren portzentaia aldakorra dela hartan inbertitutako diru kopurupean, hala eta guztiz ere, errentagarriagoa da kopuru berdina behin soilik produktu jakin batean inbertitzeak, produktu berean zenbait inbertsio egitea baino. Hau da, balidn $d=d_1+d_2$, orduan $Errenta(d,f)>Errenta(d_1,f)+Errenta(d_2,f)$.

Programazio dinamikoaren teknika erabiliz diseina ezazu algoritmo bat zehazteko lor genezakeen onura maximoa *D* hamarreko euro bagenitu (suposatu $D \leq M$).

Aurreko algoritmoa egoki ezazu onura maximo hori lortzeko jakin dezagun, gainera, zeintzuk inbertsio egin behar diren.

31) Maite eta Patxi aktoreek hamaika opari jaso dituzte entzute handia izan duen telebistako serie batean egin duten interpretazio bikainagatik. *I* opari bakoitza biontzat igorritako kaxa baten barnean dago, haren P_i pisua soilik ezaguna izanik. Opari bakoitza zer den jakiteko, kaxa ireki behar da, eta horretarako astirik ez dutela eta, opari guztiak elkarren artean banatzea erabaki dute, biok pisu berdina batzen duten kaxa multzoa bat eskuratuz. Denbora luze baten ondoren, opari sorta erdibitzea ez dute lortu. Are gehiago, euren irizpidearekin opari kaxak banagarriak ote diren ere ez dakite. Tekla informatikariari laguntza eskatu diote, tankera honetako problemak ebazten trebea baita; baina Tekla guztiz lanpeturik dagoenez eta zu haren laguna zarenez, errebote gisa problema zureganaino iritsi da. Egizu algoritmo bat Maiteren eta Patxiren irizpidea erabiliz oparien banaketa egingarria den ala ez erabakiko.

ALGORITMOEN DISEINUA

G6: ALGORITMO JALEAK

1) Kardinalitate handien duen eta elkar bateragarri diren ekintzen multzoa ekoizteko edozein hautaketa jalea ez da ona. Honela, gutxien irauten duen eta dagoeneko onartuak izan diren ekintzekin bateragarria den ekintza hautatzea eta onartzea hautaketa jalea ona ez dela frogatzeko adibide bat ematea eskatzen da. Gelditzen diren (eta dagoeneko onartuak izan diren haiekin bateragarriak diren) ekintzen artetik ekintza gutxienekin teilakatzen den ekintza hautatzea eta onartzea ere hurbilpen okerra dela adibide batez baiezta bedi.

2) *Motxilaren problema zatiekin*. Aldaera honetan motxilan objektuak osorik nahiz hauen zatiren bat sartzea onartzen da. Demagun motxilak 10 edukiera duela eta sargarriak diren objektuen pisuak (2,3,4,3,5) eta euren balioak (1,4,3,1,3) direla hurrenez hurren. Motxilan eraman litekeen irabazi maximoa zein da?

3) Alpedrete zinema aretoan hainbat pelikula proiektatu nahi dira. Emanaldi bakoitzaren hasiera ordua eta amaiera ordua ezaguna izanik, hauexek dira jaso diren emaldi eskaerak: [1,2), [3,6), [4,5), [7,10), [9,11), [11,20), [12,16), eta [15,17). Eskolan ikusitako hautaketa prozedura erabiliz, proiektatuko liratekeen pelikulak zehazta ditzazun eskatzen zaizu.

4) G=(V,E) grafo ez-zuzendu pisudun baten auzokidetza matrizea ematen zaizu:

	A	B	C	D	E	F	G	H	I
A	∞	2	∞	∞	∞	7	3	∞	∞
B	2	∞	4	∞	∞	∞	6	∞	∞
C	∞	4	∞	3	∞	∞	∞	2	∞
D	∞	∞	3	∞	1	∞	∞	8	∞
E	∞	∞	∞	1	∞	2	∞	∞	2
F	7	∞	∞	∞	2	∞	∞	∞	5
G	3	6	∞	∞	∞	∞	∞	3	1
H	∞	∞	2	8	∞	∞	3	∞	4
I	∞	∞	∞	∞	2	5	1	4	∞

Adierazpide honi dagokion grafoa marraztu. *Prim*-en algoritmoak grafo honentzat kalkulatuko lukeen *hedapen zuhaitz minimoa* kalkula ezazu; zehazki, iterazio bakoitzean Primek onartu dituen ertzak idatziz.

5) "*Pisu handien duen ertza kentzea*" prozedura hauteslearen baliagarritasuna aztertu grafoaren hedapen zuhaitz minimoa kalkulatzeko grafo konexu, ez-zuzendu eta pisu positibodunentzat. Lortutako algoritmoa eta *Prim*-ena aldera itzazu.

6) Pirata informatikari talde batek ez-legezko n kopia lortu ditu P_1, P_2,...,P_n programenak eta DVD bat argitaratu nahi du isilpean banatzeko. Programa P_i bakoitzak S_i MByte espazio behar du, eta, zorigaitzez, DVD-aren kapazitatea K izanik, hau ez da nahikoa programa denak metatzeko, $K < \sum_{i=1}^{n} S_i$ gertatzen baita:

   a) Demagun piratek diskoan metatutako programa kopurua maximizatu nahi dutela (produktu gehiago edukitzeak erakargarriagoa egingo duelakoan). "S_i *balioen ordena gorakorrean programak aukeratzen joatea*" hautaketa jale ona da? Hala balitz, algoritmo jalea egizu, bestela kontrako-adibidea eman.

   b) Demagun orain nahiago dutela diskoko edukiera gehien kontsumitzea (beste DVD batzuk grabatuko baitituzte eta horrela bolumen gutxiago geldituko zaielako grabatzeko). "S_i *balioen ordena beherakorrean programak aukeratzen joatea*" hautaketa jale ona da? Hala balitz, algoritmo jalea egizu, bestela kontrako-adibidea eman.

7) Eski irakasle batek *n* eski pare banatu behar ditu bere *n* ikasle berrien artean. Horretarako "*pertsona bati esleitu behar zaion eski parea, honen luzera pertsonaren altueratik hurbilen gelditzen den eski parea*" irizpidea erabili nahi du irakasleak. Pertsonen altueren eta eski luzeren diferentzien balio absolutuen batura minimizatzen duen eski banaketa nola egin behar du irakasleak?

8) Unibertsitateko udarako ikastaroekin batera hainbat hitzaldi antolatu nahi dira. Hamaika hitzaldi dira, eta bakoitzaren hasiera ordua eta amaiera ordua finkatuta dago. Udarako ikastaroen zuzendaritzak hitzaldietarako soilik erabiliak izango diren gela batzuk erreserbatzea erabaki du. Erreserbatu behar diren gutxieneko gela kopurua kalkulatuko duen algoritmo bat egizu eta haren denbora ordena aztertu.

9) Zehazki pisu berdina duten bi ertz dituen grafo bat emanik:

   a) Bi horietako lehenen hautatutako ertza edozein izanik, *Prim* algoritmoak hedapen zuhaitz minimo bera eraikiko luke?

   b) Aurreko atalekoa baina *Kruskal* algoritmoa erabiliz.

10) Demagun *Bib(G,v)* algoritmoa dugula, non G ertz positiboak dituen grafo bat eta bertako v erpin bat emanik v-tik G-ko gainontzeko erpinetaraino dauden distantzia minimoak itzultzen dituen. Egia da *Bib(G,v)=Bib(Kruskal(G),v)*? Gogoratu Kruskal(G) G-ko hedapen zuhaitz minimo bat dela.

11) G grafo ez-zuzendua emanik *Kruskal*-en algoritmoak *G*-ren hedapen zuhaitz minimoa kalkulatzen du *Bilatu−Bateratu* (*Union-Find*) datu-egitura erabiliz. Metodoak lehendabizi grafoaren ertzez osatutako multzoa haien pisu gorakorrean ordenatzen du, ondoren ertz bakoitza banan bana aztertzeko. Kruskalen ondorengo bariante berria aztertzea eskatzen da. Aldaera berriak meta (*Heap*) datu-egitura erabiliko du ertzak metatzeko, eta bertatik banan bana hartuko ditu ondoren iturri bertsioan bezala tratatzeko. Algoritmoa idaztea eta haren ordena kasu txarrenean kalkulatzea eskatzen da. Erantzun biak justifikatzea derrigorrezkoa da.

12) G grafo konexuko ertzetako zein *T* azpimultzok *G*-ko erpin guztiak konektatzen dituen eta ertz haien pisuen batura minimoa ematen duen kalkulatzen badakigu. *T* multzoaren kalkuluak zentzua izaten jarraitzen du oraindik nahiz *G*-ko zenbait ertzek pisu negatiboa eduki. Hala ere, kasu honetan soluzioak ez du zergatik zuhaitza izan behar. Kruskalen algoritmoa egoki bedi pisu negatibodun ertzak dituzten grafoekin ere algoritmoak funtziona dezan.

13) Erpin batetik besteetaraino dauden distantzien arteko distantzia luzeena kalkulatzeko ondorengo ideia pentsatu da: Izan bedi *Z* oso handia den zenbaki bat eta erpin auzokideen arteko distantzia berria $DBerria(u,v)=Z-D(u,v)$ formula bidez birkalkulatzen da (non u eta v edozein erpin auzokide diren). Distantzia berriak definitu ondoren, *Dijkstra*-ren algoritmoa grafo berriari aplikatuz, honek itzulitako bide motzenen distantzia, jatorrizko grafoan bide luzeenen distantziak dira. Baina, estrategia honen hanka sartzea non dago? Zergatik ez da zuzena?

14) Metropolis hiriko Trafiko Sailak zenbait kaleetan zehar konponketa lanak egitea erabaki du. Baina lan hauek aurrera emateko trafikoa etetea beharrezkoa da konponketak egiten ari diren kaleetan zehar. Trafikoa eten daitekeen kaleen zerrenda maximoa ematea eskatzen da, bidagurutze guztiak atxikigarriak izan behar dutela jakinik eta trafiko edukiera ahalik eta handien mantenduz.

15) *Dijkstra*-ren algoritmoa (grafo zuzendu bat emanik, erpin batetik beste erpinetaraino doazen bide motzenen kostua kalkulatzen duena) ikusi ondoren, hura aldaraztea eskatzen da gainera ondorengoak kalkula eta itzuli ditzan exekuzio denboraren ordena mantenduz:

    a) Zenbat bide minimo dauden erpin bakoitzeraino.

    b) Bide minimoak (hots, bideak) zeintzuk diren.

    Erantzunak arrazoi itzazu

    Arrastoa: *Dijkstra*-ren benetako algoritmoak bide motzenen kostua kalkulatzeaz at, gain- era bide propioak kalkula ditzan (erpinen sekuentziak) aski da bide horiek

metatuko dituen *BMD(1..n)* taula eranstea. *BMD(x)* gelaxkak 1etik X-rainoko bideetan justu X erpinaren aurretik bisitatu behar diren erpinen zerrenda metatuko du. Adibidez, BM D(x) = [x_1, x_2, x_3] bada, 1etetik X-rainoko motzenetan (x_1,x), (x_2,x) eta (x_3,x) bisitatzen diren arkuak dira.

16) Demagun grafo zuzendu azikliko bat dugula, non haren arku bakoitzak pisu ez-negatibo bat erlazionaturik duen. Helburua, abiapuntu batetik beste erpinetaraino doazen bideen arteko bide luzeenen distantziak zenbatekoa den kalkulatzea da. Zuzena litzateke ondorengo hautaketa jalea erabiltzea algoritmo jale batez problema ebazteko?

Izan bedi S onartutako erpinen multzoa (hasieran S={Abiapuntua}). Pauso bakoitzean bide berezi luzeena duen eta e∉S den erpina onartuko du hautaketa jaleak.

(Bide bereziaren kontzeptua *Dijkstra*-ren algoritmoan ikusitakoa da.)

17) Demagun G grafo azikliko arkuen pisuak 0 edo balio handiagoa dutela. *Dijkstra*-ren algoritmoan, izan bedi *w* soluzio *S* multzotik at dagoen erpin bat *S*-ri *v* erpin berri bat erantsi ondoren. Ba al lezake abiapuntu erpinetik *w*-rainoko bide berezi motzen batek *v*-tik pasa ondoren *S*-ko beste erpin batetik pasatzea? Erantzuna matematikoki argudiatu behar da.

18) Tokiz aldatu nahi dugun *L* litroko depositu bat dugu. Duen bolumenagatik, toki aldaketa hutsik dagoelarik egin behar da. Deposituko edukiaren garraiorako beste *N* depositu dituen tren bat erabiliko da. Treneko deposituek behar haina edukiera duten arren, txartuta daude eta garraiatzen duten likidoaren zati bat garraioan galtzen dute. Depositu bakoitzak galtzen duen likidoa, daramaten kantitatearen menpe dago. *G(1..N,1..L)* matrize bat dugu, non *G(i,k)* balioak *i* deposituan *k* litro sartuez gero galduko liratekeen litroen estimazioa adierazten duen. *L* litro garraiatzerakoan gutxienez galduko diren litroen estimazioa kalkulatu nahi dugu bai eta depositu bakoitzean eraman behar diren litro kopurua ere. Proposatutako soluzioaren denbora-ordena eta erabilitako memoria espazio estra kalkula ezazu.

19) Huffmanen kodeketa:

• Zuhaitz bitar ez-beteak aurre-kodeketa optimoei ez dakiekeela froga bedi.

• Aurre-kodeketa optimoen zuhaitzen kostu guztia ere barne adabegi bakoitzaren bi haurren maiztasunen konbinazioa eginez kalkula daitekeela froga bedi.

• Demagun 8 bit erabiliz kodetzen diren karaktereez osaturik dauden datuen fitxategi bat dugula. 256 karaktere hauen maiztasunei dagokienez, ezaguna da maiztasun handiena maiztasun txikienaren bikoitza baino txikiagoa dela. Froga bedi Huffmanen kodeketa erabiltzea 8 biten luzera finkoko kodeketa arrunta baina eraginkorragoa ez dela.

20) Eskiko irakaslea batek n pare eski bere ikasle berrien artean banatu behar ditu. Eskiko klubaren erregelak jarraituez gero, ikasle bakoitzak bere altuerarekin bat datorren eski parea jaso beharko luke. Irakaslearen problema *n* eski pareak bere *n* ikasle berrien artean ahalik eta hobekien banatzea da. *Ikasle bakoitzaren altueraren eta honi esleitutako eski parearen luzeraren arteko diferentziaren balio absolutuen batura minimizatzea* da irakaslearentzat asignaziorik onena.

Problema ebazteko nahikoa izango da ikasle baxuenari eski pare motzena esleitzea eta modu berean jarraitzea gainontzeko ikasleekin. Estrategia zuzena dela froga bedi.

21) *Midas* irakaslea bere autoa Irunetik Tarifara ari da gidatzen *X* errepidetik. Bere gasolio depositua beterik dagoela habiatzen bada, Midas-ek *n Km* gida ditzake. Bestalde bere errepide mapa eguneratuan zerbitzuguneak markaturik datoz, gainera bere bideko zerbitzuguneen arteko distantzia kilometrotan azaltzen du. Irakasleak ahalik eta geldiune kopuru txikiena egin nahi duela eta, Midas irakasleak zein zerbitzugunetan gelditu beharko lukeen adieraziko duen algoritmo eraginkor bat idaztea eskatzen da. Erabilitako estrategiak beti soluzio onena emango duela froga bedi.

22) ***K*** euro sakelan izanik azokara goaz. Honekin batera eros ditzakegun ***m*** produktuen zerrenda daramagu. ***i*** produktu bakoitzeko ($1 \le i \le n$) p_i prezioa eta e_i erabilgarritasun balioa (biak osoko positiboak) ezagunak ditugu. Produktu bakoitzeko gehienez 3 produktu eros ditzakegu. Gainera, eguneko eskaintzari esker, produktu beraren bigarren unitatearen salneurria 1 euro gutxiago kostatuko zaigu eta hirugarren unitatea, aldiz, 2 euro gutxiago. Erosketaren erabilgarritasuna erositako produktu bakoitzaren erabilgarritasunaren batura dela jakinik, *Programazio Dinamikoaren* teknika aplikatuz algoritmo bat idatz ezazu gehienez K euro azokako produktuetan gastatuz lor dezakegun erabilgarritasun maximoa mugatzeko bai eta erosi behar den **elementuen zerrenda** ere. Proposatutako soluzioaren denbora-ordena eta erabilitako memoria espazio estra kalkula ezazu.

23) Izan bedi M_1, ..., M_n *n* matrizez osaturiko sekuentzia, non M_i matrizearen dimentsioak $D_{i-1} \times D_i$ den $\forall i$ $1 \le i \le n$. Ondoren, matrize-sekuentziako matrizeen arteko biderkatzea egiteko estrategia jale bat proposatzen da:

Pauso bakoitzean, gelditzen diren matrizeen dimentsio handiena hartu bedi eta dimentsio hori konpartitzen duten bi matrize auzokideen arteko biderkatzea egin bedi.

Estrategia honek faktoketa-ordena optimoa ematen duela ikus bedi.

a) Algoritmoaren exekuzio-ordena zein da? (Faktorketa egiteko soilik, benetako biderkaketa egin gabe)

b) Edo estrategiak beti faktorketa optimoa kalkulatzen duela froga bedi (hots, seguru eta induktibo propietateak frogatu) edo aurkako adibide bat eman bedi.

ALGORITMOEN DISEINUA

G7: BACKTRACK: Atzera jotze inteligentea

1) L balioa ematen digute bai eta $M=\{z_1, z_2,\ldots,z_n\}$ zenbakizko multzoa ere. Backtrack teknika erabiliz, M-ko zenbat azpimultzo dago osagaien batura zehazki L dena?

   Zuhaitza irudikatuz hasi, adarketa identifikatu, behar diren aldagaiak identifikatu, soluzio partziala soluzioa noiz den (hostoa) identifikatu, kima posibleak aztertu eta euren eragina eta azkenik algoritmoa idatzi.

2) *Irakasleen kontratazioa*. Ikastegi batek *IK* ordu irakasteko irakasleria kontratu behar du. Horretarako, N klase desberdinetako kontratuak egin ditzake: i klaseko irakasleak O_i ordu-blokea irakats dezake gehienez eta bloke osoagatik beti P_i euro ordainduko litzaizkioke (hots, kontratatuko O_i ordu guztiak ez irakastea onartzen da). Baina, ikastegiak ez ditu M euro baino gehiago ordaindu nahi.

   Backtack teknika erabiliz diseina ezazu algoritmo bat, irakasleen kontratazioaren problemak soluzioa duen mugatzeko, eta soluzioa balu, problema ebazten duten kontratuen bilduma emango lukeena. Saiakeren zuhaitza irudikatu azalpenak emanez (adarketa, kimak, parametrizazioa, soluzioaren itxura,…)

3) Z zenbaki bat emanik, kopuru hori batzen duten eta goranzko ordenan dauden osoko positibo zenbakizko zerrenda guztiak kalkulatzen dituen backtrack algoritmoa egizu. Adibidez, $Z=6$ balitz, algoritmoaren irteerak *[1,2,3], [1,5], [2,4]* eta *[6]* zerrendak eman beharko lituzke.

4) R zenbakien karratuen baturaz X zenbakia lor litekeen ala ez mugatzen duen algoritmoa egizu.

5) *Alpedrete* zinema aretoan hainbat pelikula proiektatu nahi dira. Emanaldi bakoitzaren hasiera ordua eta amaiera ordua ezaguna izanik, hauexek dira jaso diren emaldi eskaerak: [1,2), [3,6), [4,5), [7,10), [9,11), [11,20), [12,16), eta [15,17). Eskolan ikusitako hautaketa prozedura erabiliz, proiektatuko liratekeen pelikulak zehazta ditzazun eskatzen zaizu.

6) 8x8ko xake taula batean elkar-eraso gabe 8 erregina kokatzeko posizio-konbinazio bat edo denak mugatu.

7) *Erregearen problema*. Problema honakoa da: xake taulako *(i,j)* gelaxkan errege bat kokatu da, kalkula ezazu –existitzen badu- errege-mugimenduen sekuentzia bat non *(i,j)* gelaxka abiapuntutik hasita taulako gelaxka guztiak bakarrik behin zeharkatzen dituen. Erregearen problema backtrack teknika erabiliz gara ezazu. Algoritmoaren analisia egizu. Iruzkina: xakeko erregea horizontalki, bertikalki nahiz diagonalean baina alboan dagoen gelaxkara soilik mugi liteke.

8) *Xake-zaldia*. Zaldia *NxN* gelaxka dituen xake moduko taula bateko (x,y) gelaxkaren batean dago kokatua. N^2-1 zaldi-mugimenduetan taulako gelaxka guztiak behin soilik bisita litezkeen erabaki behar duen algoritmoa egizu backtrack teknika erabiliz.

   Koordenatuen kalkuluak eta idazkera laburtzeko, ondoren azaltzen zaizun funtzioa erabil dezakezu. M zenbakiak zaldiak egin ditzakeen 8 mugimenduen arteko bat adierazten du. Iparretik hasi eta ordu-orratzen zentzua jarraituz, 1..8 zenbakitzen ditugu. Honela *(i,j)* koordenatuan balego zaldia, m aukera egingo balitz, jomuga gelaxkaren koordenatuak *f(i,j,m)* espresioak emango luke.

   a) Backtrack algoritmoaren estrategia ulergarria gerta dakizun, saiakeren zuhaitza irudikatu eta azaldu.

   b) Saiakeren zuhaitzean deskribatutako estrategia inplementatzen duen backtrack algoritmoa idatz ezazu.

9) *Bikoteen agentzia*. Bikoteen agentzia batean n gizonezkoen eta n emakumezkoen preferentzien *G(1..n,1..n)* eta *E(1..n,1..n)* informazio taulak dituzte, hurrenez hurren. i emakumezko bakoitzeko, *E(i,j)* balioak, *i* emakumezkoak *j* gizonezkoagatik duen preferentzi gradua adierazten du. Paretsu, *x* gizonezko bakoitzeko, *G(x,y)* balioak *x* gizonezkoak *y* emakumezkoagatik duen preferentzi gradua adierazten du.

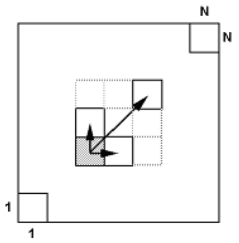
   Gizon-emakumezko parekatze bijektibo bat proposatuko dion backtrack algoritmo bat egizu bikoteen agentziarentzat, bikoteetako preferentzien biderketen batura minimizatuko duena.

10) *N* objektuk p_1, p_2, ..., *pn* pisuak eta b_1, b_2, ..., *bn* balioak erlazionaturik dituzte hurrenez hurren. Bestalde, *E* edukiera duen motxila bat ere badugu. Helburua, motxilaren edukiera gaindituko ez duten objektuen azpimultzo bat bilatzea da haien balioen batura maximizatuz bai eta motxilaren etekin zehatza ere azpimultzo horrentzat. Backtracking eskema erabiltzen duen soluzioa ematea eskatzen da.

11) *E* edukiera duen motxila eta *N objektu klase* eskuragarri ditugu, klase bakoitzeko behar adina objektu edukiz. Bestalde, i klaseko ($1 \le i \le N$) objektuaren balioa eta pisua dakizkigu, b_i eta p_i izanik. Motxilaren edukiera gainditu gabe, objektuak sartuz, motxilaren irabazia maximizatzen duten objektuen azpimultzoa mugatu nahi da bai eta motxilaren etekin zehatza ere azpimultzo horrentzat.

Zertan eragingo luke esango baligute *i* klase bakoitzeko m*i* objektu ditugula?

12) (X_i,Y_j) jatorri puntutik (X_n,Y_n) helmuga puntura zenbat bide posible existitzen den mugatzen duen algoritmo bat idatzi bedi. Puntuen koordenatuak zenbaki arruntak direla suposatu. Bestalde, (X,Y) puntu batetik beste puntu batera joateko eman daitezkeen pausoak $(X+1,Y)$ eta $(X,Y+1)$ bakarrik izango dira. Algoritmoaren exekuzio-denbora aztertu.



13) Midas irakaslea bere autoa Irunetik Tarifara ari da gidatzen X errepidetik. Bere gasolio gordetegia beterik dagoela abiatzen bada, Midas-ek *N* kilometro gida ditzake. Bestalde bere errepide mapa eguneratuan zerbitzuguneak markaturik datoz, eta bere bideko zerbitzuguneen arteko distantzia ere azaltzen du kilometrotan. Irakasleak ahalik eta geldiune kopuru txikiena egin nahi duela eta, Midas irakasleak zein zerbitzugunetan gelditu beharko lukeen adieraziko duen algoritmo eraginkor bat idaztea eskatzen da. Erabilitako estrategiak beti soluzio onena emango duela froga bedi.

14) Demagun $N+1$ zenbaki osoko positibo ditugula, w_i $(1 \le i \le N)$ eta K.

   a) Optimizazio problema: $w_1,...,w_n$ osokoen azpimultzo desberdin guztien artetik zein azpimultzo da objektuen batura K-tik hurbilen egonik K kopurua gainditzen ez duena?

   b) Erabaki problema: existitzen du w1,...,wn objektuen azpimultzorik balioen batura zehazki K dena? Azpimultzoak existitzen badu, gainera, itzuli azpimultzoa ere.

15) Demagun 1 bolumeneko edukiera duten nahi adina saski (bagoi) dugula eta $b_1,b_2,...,b_N$ bolumena duten N objektu ditugula, non $\forall i(1 \le i \le N \leftarrow 0 < b_i \le 1)$ diren.

   a) Optimizazio problema: Objektu guztiak paketatzeko behar diren saski kopuru txikiena zein den mugatu.

   b) Erabaki problema: Sarrerako datuez gain, gainera K osokoa emanik, objektu guztiak K saskitan paketa litezke?

16) Hamilton zirkuitua/zikloa. Hamilton zikloa grafo zuzendu/ez-zuzendu batean grafoko erpin bakoitza zehazki behin zeharkatzen duen ibilbide/zikloa da.

   a) Erabaki problema: Sarrerako grafoak Hamilton ziklorik du?

17) Saltzailearen problema (Traveling salesperson/Minimun tour). Grafo pisuduna sarrerako datua da.

   a) Optimizazio problema: Pisu minimoa duen Hamilton zirkuitua mugatu. Saltzailearen problema izenez ezaguna da enuntziatu hau. Saltzaileak eskualde bat bisitatzen duenean orotara egin beharreko distantzia minimizatu nahi baitu. Beste aplikazio praktikoak dira: kamioien bideratzeak zaborrak biltzeko eta paketeen banaketak.

   b) Erabaki problema: Grafoaz gain *K* osokoa dugu. Hamilton zirkuiturik badago gehienez orora *K* pisua duena?

18) Demagun *G* grafoa dugula bai eta M osoko positiboa ere. Jakinik gehienez M margo kolore erabil ditzakegula eta jakinik auzokide erpinak kolore desberdinez margotuta egon behar dutela, *G*-ko erpinak margo ote ditzakegun erabaki nahi dugu. Adibidea: Grafoa Koloratzearen aplikazio praktiko bat: Demagun unibertsitate batetako azken azterketak aste betean egiten direla eguneko hiru azterketa desberdin antolatuz eta guztira 15 denbora gelaxka betez. Ikasturte batzuetako azterketak, adibidez, Fisika I eta Kalkulua I, ordu desberdinetan antolatu behar dira irakasgai bietan dauden ikasle asko baitago. Izan bedi V irakasgaien multzoa eta izan bedi E ordutegi desberdinetan ospatu behar diren irakasgai pareen multzoa. Hau dena jakinik, 15 denbora gelaxkatan azterketa guztiak antola litezke arazorik gabe, baldin eta soilik baldin $G=(V, E)$ grafoa 15 koloretan margotu baliteke.

19) *Atazen banaketa agente artean*. Demagun N agente ditugula eta N ataza. i agenteari $(1 \le i \le N)$ j ataza $(1 \le j \le N)$ esleitzen badiogu, honen egiteak K_{ij} kostua du. Kostu guztiak K matrizean ditugu. Helburua, agente bakoitzari ataza desberdin bat esleitzea da ataza guztiak burutu daitezen, baino atazen burutzearen kostu guztien batura minimoa eginik.

20) Ataza banaketa zigorrekin (penalizazioak). Demagun N ataza egikaritu behar direla sekuentzialki: hots, une bakoitzean A_i ataza bakar bat egikari liteke. Bestalde, A_i ataza bakoitzeko ematen digute $(1 \le i \le N)$:

   a) *ti* denbora: A_i atazaren exekuzioak behar duen denbora.

   b) *bi* denbora: A_i ataza beranduen egikaritzen has litekeen denbora zigorrik gabe.Denbora hau lehenengo ataza exekutatzen hasten den denboratik neurtu da.

   c) *pi* kostua: A_i ataza *bi* denbora baino beranduago egikaritzen hasteagatik jasan beharreko zigorra.

   Denbora eta zigor balio guztiak osoko positiboak direla suposatuko dugu. Atazen banaketa denboran zehar π permutazio bat da {1, 2, 3, ..., N } osagaiena, non $A_{\pi(1)}$ lehenen exekutatzen den ataza adierazten duen, $A_{\pi(2)}$ ondoren egikaritzen dena eta abar.

Atazen banaketa konkretu batek orotara lukeen zigorra ondorengo ekuazioak jasotzen du:

$$Z_\pi = \sum_{A=1}^{N} \left[if\left(t_{\pi(1)} + \ \ldots + t_{\pi(A)}\right) > b_{\pi(A)} then\ Z_{\pi(A)}\ else\ 0 \right]$$

d) Optimizazio problema: Kalkulatu zein den zigor minimoa ekoizten duen atazen banaketa bai eta banaketa horri legokiokeen zigorra guztira.

e) Erabaki problema: Sarrerako datuez gain, *K* osoko positiboa emanik, existitzen du atazen banaketarik $Z_\pi \leq K$ duena?

21) Akademia Antolakuntzarako Dekanorde batek irakasgaien azterketak egingarriak diren egunetan antolatu nahi ditu. Kanpoko zenbait faktorek (ikastaroak, ikastegi kanpoko gelak, etab.) egunero erabilgarri dauden eserlekuak konstantea ez izatea dakarte.

Egunetan zehar azterketa guztien antolaketen aukera guztiak ekoizten dituen algoritmo bat diseina diezaiogula eskatu digu Dekanordeak, aukera desberdinak ikasleekin eztabaidatzeko asmoz.

Irakasgai bakoitzak behar dituen eserleku kopurua ezaguna da. Azterketa guztien iraupena 2 ordukoa da eta egunean zehar 8 ordu ditugu azterketak egiteko. Bestalde, 15 egunetan burutu behar diren azterketak 50 dira.

Gainera, Matrikulan izeneko matrize bat daukagu, non irak irakasgaian ikas ikaslea matrikulaturik dagoenen true jasoa duen, *Matrikulan(irak,ikas)=true*, eta *false* bestela.

Esleipena onargarria izan dadin, irakasgai bakoitzaren azterketak behar dituen eserleku adina eserleku erabilgarri egotea derrigorrezkoa da eta elkar bateraezinak diren irakasgaien azterketak ezingo dira egun berean ospatu (hau da, egun berean ikasle batek bi azterketa izatea onartezina da).

22) Moldeagarri diren *n* objektuen bilduma dugu, bakoitzaren v_i bolumena ezaguna delarik, $i=1,\ldots,n$. Objektu hauek E edukiera duten ontzietan paketatu behar dira. Jakinik objektuak ezin direla zatikatu, paketatze optimoa kalkulatuko duen algoritmoa egin ezazu, hots, erabilitako ontzi kopurua minimizatzen duena.

23) Fanfanisflaneko errepublikan puntako teknologia duten n faktori zabaldu berri dira, eta haien instalazio informatikoak eta haiek zuzenduko dituzten pertsonala behar da. Gobernu Gorenak *n* makinak esleitu ditu eta titulatu berri diren beste hainbeste teknikari, fabriken artean antolatu behar direnak. Makina guztiak ez dira egokiak fabrika guztietan, eta hori jasotzeko *egokiaDa?(m,f)* funtzio boolearra daukagu, *m* makina *f* fabrikan egokia den ala ez jasotzen duena. Era berean, teknikari guztiek ez dituzte makina guztiak ezagutzen, ondorioz, *ezagutzenDu?(t,m)* funtzio boolearrak *t* teknikariak *m* makina jasotzen duen ala ez jakinarazteko erabil ahal izango dugu. Azkenik, edozein teknikarik ez du edozein fabrikan lan egin nahi, zenbait fabrika armen industriarekin erlazionatuta baitaude, horretarako *onartukoLuke?(t,f)* funtzioa erabili ahal izango dugu,

non *t* teknikaria *f* fabrikan lan egiteko prest legokeen ala ez jasotzen duena. Helburua, esleipen onargarri bat kalkulatzea da, balego, non fabrika bakoitzari euren eginkizunetarako egokia den makina bat esleitzen dion bai eta teknikari bat ere, makina hori ezagutzen duena eta fabrika horretan lan egiteko prest legokeena.

24) *N* langile eta *N* ataza ditugu. I langilearen kualifikazioa J ataza egiteko *Kualif(i,j)*>0 da. Ondorengoak kontuan edukiz:

a) X_{ij} aldagaiak 0 balio du I langileari ez bazaio J ataza esleitzen; eta 1 bestela.

b) *J* atazarentzat $X_{1j}+\ldots+X_{nj}=1$ betetzen da; ondorioz, ataza bakoitza langile batek egingo du.

c) *I* langilearentzat $X_{i1}+\ldots+X_{in}=1$ betetzen da; ondorioz, langile bakoitzak ataza bat egingo du.

Helburua, langileen bidezko atazen burutzeak kualifikazio maximoz egitea da, hau da, maximizatzea $\Sigma i\ \Sigma j\ Kualif(i,j)\ X_{ij}$ backtrak teknika erabiliz.

ALGORITMOEN DISEINUA

Az8: Denetarik

1) n≥1 zenbaki naturala triangeluarra dela diogu baldin eta 1ean hasi eta jarraikiko zenbakiz osaturiko ordena gorakorreko sekuentzia ez-huts baten baturaren berdina bada. Ondorioz, lehenengo bost zenbaki triangeluarrak 1, 3=1+2, 6=1+2+3, 10=1+2+3+4 eta 15=1+2+3+4+5 dira. Idatz ezazu programa bat non emaniko zenbaki osoko positiboa zenbaki triangeluarra den ala ez erabakitzen duena. Programaren eraginkortasuna O(n)-ren barnekoa izan behar du eta erabilitako memoria espazio estra konstantea. Zure soluzioaren ordena aztertu.

2) Izan bedi V[1..n] erregistroen bektorea. Erregistroek kolorea izeneko eremu bat dute, non gorria, berdea edo urdina balioak har ditzaketen. Denbora lineala beharko duen algoritmo bat egizu V bektoreko balioak berrantolatzeko, kolore berdineko erregistroak jarraian gera daitezen. Zehazki, hasierako posizioetan gorri kolorekoak jarriko dira, ondoren berde kolorekoak eta azkeneko posizioetan urdinak.

3) Ordenaturik dagoen V(1..n) bektorea emanik eta Z eta S bi balio (non Z≤S), Z eta S balioen arteko balioz osaturiko V-ko azpibektorearen bi indizeak mugatzeko algoritmo bat egizu bilaketa dikotomikoaren ideia erabiliz. Soluzioak kasu txarrenean O(lg n) ordena duela frogatu. Adibidez, sarrera V=(3, 3, 3, 5, 5, 5, 7, 8, 8, 10, 12, 12, 12, 12, 12); Z=5 eta S=8 denean, irteerak (4,9) izan behar du.

4) *Gr=(ERP, AR)* grafo zuzendu pisudunak komunikazio sare bat adierazten du. *(u,v)∈AR* arkuaren pisua f(u,v) da, bere balioa *0≤f(u,v)≤1* tartean egonik, *u*-tik *v*-raino komunikazio kanalaren fidagarritasuna adierazten du, hots, *u-v* kanalaren ez hutsegitearen probabilitatea. Honela, *f(u,v)=0* denean, kanala ez dela batere fidagarria adieraziko du, eta aldiz, *f(u,v)=1* denean, komunikazio kanala erabat fidagarria dela adieraziko du. Demagun Xetik Yra doan kanalaren fidagarritasunaren huts egitearen probabilitatea kanala osatzen duten zatien probabilitateen biderkadura dela. Adibidez, *[x,y]* bideak *(x,z)* eta *(z,y)* bi kanalak zeharkatzen dituen bidea bada, *[x,y]=(x,z)(z,y)*, orduan *f[x,y]= f(x,z)*f(z,y)*.

   A eta *B* erpinen arteko bide fidagarriena kalkulatzeko algoritmo eraginkor bat egizu. Soluzio eraikitzeko Dijkstraren soluzioan inspira zaitez.

5) Europako hainbat lurraldeetan "koipezko sukarra" epidemia pairatzen ari da. Gaixotasuna kutsatuta dauden abereekin kontatu zuzena edo zeharkakoa izanagatik

hedatzen da eta gainera, guztiz kutsakor ,dela deskubritu dute. Infekzioa kontrolatzeko asmoz, granja guztietatik informazio bildu dute osasun autoritateek, G=(g1,g2,...gn) izendatu direnak. Gainera, granjen artean egin diren abere salmentak eta hauen datak erregistratu dira.

Orain arte birusa granjen IG izeneko azpimultzo kaxkarrean detektatu da. Autoritateak kutsatuta egon litezkeen granjak identifikatu nahi ditu, berrogei eguneko itxialdia ezartzeko haietan. Kontuan izan behar da granja bat kutsatuta egoteko arriskua duela, baldin eta

- kutsatuta zegoen granja batetik abereak erosi baditu, edota
- kutsatuta egoteko arriskua zuen granja batetik abereak erosi baditu (hots, granja saltzailea kutsatua izan ostean saldutako abereak granja eroslea kutsa lezake).

Koarentenan aldarrikatu behar diren granjak zehaztuko dituen algoritmoa diseina eta azter ezazu.

6) Izan bedi A *n* osokoen taula, n>0 izanik eta izan bedi *Batu*? funtzioa honako hau:

```
function Batu?(A,1, i )) return boolean is   -- i>0
   if i=1 and A(1)/=0 then return false
   elsif A(i)= Batukaria(A,1,i-1) then retun true
   else return Batu?(A,1,i-1))
```

non

```
function Batukaria(A,1, k )) return integer is
   b:=integer:=0
   for j:= 1 to k loop b:=b+A(j) end loop
   return b
```

Batu?(A,1,n) espresioaren ebaluazioak *true* itzultzen du A taulako osagai bat aurreko osagaien baturaren berdina den kasuan; eta *false*, baldintza hori betetzen duen osagairik existitzen ez duenean A taulan. Algoritmoaren eraginkortasuna aztertzea eskatzen zaizu. Erabil ezazu beste teknika bat problema bera eraginkorrago ebazteko. Soluzio berriaren eraginkortasuna aztertu.

7) Garraio enpresa batek *M* kaxez osatutako bidalketa garrantzitsu bat onartu du. Horretarako, *N* klaseetako ibilgailuak ditu. *I* klaseko ibilgailuek K_i kaxentzako kapazitate maximoa dute. *I* klaseko ibilgailu baten garraioak P_i prezio konstantea du, (ibilgailua beterik egon ala ez, eta hurbil joan ala ez). *M* kaxak kostu minimoan garraiatzeko **ibilgailuen antolaketa optimoa** kalkula ezazu programazio dinamikoa erabiliz. Proposatutako soluzioaren denbora-ordena eta erabilitako memoria espazio estra kalkula ezazu. Oharra: Garraioaren kostuaz gain, ez ahaztu ibilgailu mota bakoitzeko zenbat ibilgailu esleituko diren jakin nahi dela. Proposatutako soluzioaren denbora-ordena eta erabilitako memoria espazio estra kalkula ezazu.

8) Goranzko ordena jarraituz ordenatutako gizakien izenak dituzten *G1(1..N)* eta *G2(1..N)* bi bektore ditugu. Aldi berean bi bektoreetan agertzen diren gizakien izenak kalkulatuko dituen programa idatzi behar da. Gizakien bektoreen luzera *n* denean, soluzioak har dezakeen denbora-ordena $T_t(n) \in \mathbf{o}(n \lg n)$ baina hobea izatea eskatzen da. Oharra: jatorrizko bektoretan izen errepikatuak baleude ere, ebakiduran ez lukete egon beharko.

9) Ausazko zenbakiek hamaika aplikagunetan dute: simulazioak, jokoak, segurtasun digitala, … Konputagailu bidez ekoizten diren ausazko zenbakiak ez dira ausazko puruak. Ausazko aizunak (pseudo-ausazko) deritze eta formularen batetik ekoizten badira ere, estatistikoki aurretik ezin dira jakin. Ausazko zenbakiak modu sinplean ondorengo formula bidez konputa litezke. a, b eta m balioak emanik

$$x_{i+1}=(a*x_i+b) \bmod m$$

hurrengo ausazko zenbakia sortzen du.

a) Programazio dinamikoaren teknika bidez bai iteratiboki bai errekurtsiboki (memoriadun funtzioak erabiliz), lehenengo k ausazko zenbakiak itzuliko dituen algoritmoak egin itzazu, a, b, m eta x_0 hazia datuetatik.

```
Ausazko ((25,34,128),51,7)=[51,29,119,65,123,37,63]
   non a=25, b=34, m=8 eta x0=51 K =7
```

b) Aurreko ataleko soluzioen konplexutasuna aztertu bai eta erabilitako memoria espazio estra ere.

10) Demagun liburutegiaren arduraduna zarela. Jendeak egunero liburuak erabili ondoren bere lekuetan kokatu behar ditu, baina batzuetan nahasten dira eta toki okerrean uzten dituzte. Liburutegi osoaren tamainarekin alderaturik, gaizki kokaturik suertatzen diren liburuen kopurua oso txikia da, eta gainera haien lekutik dezente gertu gelditzen ohi dira. Eguna bukatzean, liburu guztiak bere tokietan egotea lortu behar duzu.

a) Klasean ikusitako algoritmoetatik zein uste duzu egokiena denik problema honi aplikatzeko? Zergatia azaldu, zure aukera beste algoritmoak baino hobea dela egiaztatuz (gutxienez beste hiru algoritmorekin konparatu behar duzu)

b) Benetako liburutegietan hartutako liburuak birkokatzea erabiltzaileei ez zaie onartzen, eta arduradunek beraiek lan hori egitea nahiago izaten dute. Horren zergatia esplikatu (liburutegia ordenatuta mantentzearen kostuan oinarrituz).

11) X osagaia *S(1..N,1..M)* taula batean dagoen erabakiko duen programa egizu, non taulako lerroak eta zutabeak bektore ordenatuak diren. Soluzioaren *T(n,m)* exekuzio denbora eta ordena bai eta erabilitako *MEE(n,m)* mugatu.

12) G grafo ez-zuzendua emanik Kruskalen algoritmoak G-ren hedapen zuhaitz minimoa kalkulatzen du Bilatu-Bateratu (*Union-Find*) datu-egitura erabiliz. Metodoak lehendabizi grafoaren ertzez osatutako multzoa haien pisu gorakorrean ordenatzen du, ondoren

ertz bakoitza banan bana aztertzeko. Kruskalen ondorengo bariante berria aztertzea eskatzen da. Aldaera berriak meta (HEAP) datu-egitura erabiliko du ertzak metatzeko, eta bertatik banan bana ertzak hartuko ditu ondoren iturri bertsioan bezala tratatzeko. Algoritmoa idaztea eta haren ordena kasu txarrenean kalkulatzea eskatzen da. Erantzun biak justifikatzea derrigorrezkoa da.

13) Zirkuiting enpresak zirkuitu digitalak diseinatzen ditu eta euren propietateak aztertzeko programak bilakatzen hasi dira. Oraingoan, ondorengo osagaiak dituzten zirkuitu sinpleak aztertzeko programa bilakatzea eskatu du: pila baten polo positiboari eta polo negatiboari konektatutako kable parea, kableak konexio puntuak lotuz, bai eta kableak bonbillaren bi kontaktu puntuetara lotuak. Programak emaniko zirkuituak ondorengo hiru propietateko zein propietate betetzen duen itzuli behar du:

a) Bonbilla piztuko da

b) kortozirkuitua gertatuko da, pilaren polo positibotik negatibora eta bonbillatik pasa gabe korrontea iritsiko delako.

c) Bonbilla ez da piztuko korrontea hura zeharkatzen ez duelako, baina kortozirkuitua ere ez da gertatuko korrontea polo negatiboraino iristen ez delako.

14) Honako definizio errekurtsiboa ematen digute:

$$f(n,m)=\begin{cases} 0 & m=0 \\ n & m=1 \\ f\left(n,\left\lfloor \frac{m}{2} \right\rfloor\right)+f\left(n,\left\lceil \frac{m}{2} \right\rceil\right) & m>1 \end{cases}$$

- Begi bistakoa da programazio errekurtsibo inozoaz (puruz) kodetuko balitz, kalkuluak errepikatuko liratekeela. Hauek ekiditeko, programazio dinamikoaren teknika erabiliz inplementa ezazu bai iteratiboki bai memoriadun funtzioak erabiliz, hots, bi moduetan.
- Aurreko ataleko soluzioen konplexutasun analisia egizu. Batuketa kopuruan diferentziarik dagoenik nabarmentzen al duzu?

15) *n* pertsonen artean, pertsona bat ospetsua da baldin eta denek hura ezagutzen badute baina hark inor ezagutzen ez badu. Pertsona ospetsurik existitzen den identifikatzeko ondoren galdera soilik uzten digute egiten: "Barkatu, hango pertsona hura ezagutzen duzu?" Galdera n pertsonetako edozeini egin badiezaiokegu eta jasotako erantzunak beti zuzenak direla suposatzen badugu, zein da egin egin ditzakegun galdera kopuru txikiena n pertsonen artean, haien arteko norbait ospetsua den edo ez mugatzeko?

16) *n* teorikok esamesak elkarraldatu nahi dituzte. Honela, telefono deiak 2 teoriko artean gertatzen badira, eta bi teorikok hitz egiten dutenean, biok dakizkiten esamesa guztiak elkarraldatzen badituzte, zenbat telefono dei egin beharko dira gutxienez n teorikoek esamesa guztiak jakin ditzaten?