UPV/EHU—Informatika Ingenieritza
DEA-II 1 PARTZIALA (2011-iii-8) - E taldea

Deiturak	1	2	3	4	5	

1. (2 pt) *Egiazkoa* edo *faltsua* balioekin osatu ondorengo taulako gelaxkak, behar den bezala, azkeneko zutabean kostu asintotiko txikien duen funtzioa adieraziz:

f(n)	g(n)	f(n)∈O(g(n))	f(n)∈Ω(g(n))	f(n)∈Θ(g(n))	Onena
$\sqrt{n} * (\lg n)$	n				
$4^{\log_2 n}$	$2n^2$				
$n\sqrt{n}$	$3n^{1.4}$				

2. (2 pt) A(i..j) bektoreko k balio txikiena *KHautaketa*(A,i,j,k) algoritmoak kalkulatzen duela ikusia dugu. Aplika ezazu *KHautaketa* k=7 eta ondorengo bektoreari:

A= « 46, 27, 2, 75, 83, 9, 51, 6, 58, 55 »

Deien ondorioz suertatuko balioekin osatu tauletako gelaxkak. Gainera, deien burukoen balio konkretuak zehaztea ere eskatzen zaizu (hots, azpitaularen hasiera eta bukaera indizeen balioak hala nola zenbat garren balio txikiena bilatzen den).

KHautaketa (A, 1, 10, 7) *Banaketa*(A, 1, 10, BP) // non BP= }

KHautaketa (A, , ,) *Banaketa*(A, , , BP) // non BP= }

KHautaketa (A, , ,) *Banaketa*(A, , , BP) // non BP= }

3. (2 pt) Ondorengo algoritmoaren denbora ordena kalkulatu.

```
proz AzterNazazu (M) is
   MetaEraiki(M);
   b ← (M(1) + M(M.last))/2;
   N ← M.last;
   for  i  in  1..N  loop
      b ← b + (-1)^i M(i);
      M.last← M.last+1;  M(M.last)←b;  Azaleratu(M, M.last);
      b ←M(1);  M(1)←M(M.last);  M.last←M.last-1;  Hondoratu(M,1);
   end loop;
end proz.
```

4. (2 pt) Zenbaki osokoen bektore bat emanik, osagai handiena eta txikiena kalkulatzen duen eta *2n-3* konparazio egiten dituen soluzioa bilatzea erraza da. Bestalde, badakigu problema hori ebazten duten algoritmoek Θ(n) denbora ordena dutela. Oraingoan, *zatitu eta irabazi teknika* erabiliz eta *2n-3* baino konparazio gutxiago burutzen dituen soluzio bat gara dezazula eskatzen dizugu. Zure algoritmoak egiten dituen konparazio kopuru zehatza ere kalkula ezazu.

5. (2 pt) Grafo zuzendu batean, *jatorri* eta *jomuga* erpinak seinalatu ditugu. Gainera erpin *Debekatu*en multzo bat ere badugu. Egizu algoritmo bat erabaki dezan jatorritik jomugara bidea dagoen ala ez, grafoko arkuak zeharkatuz baina erpin Debekatuak zeharkatu gabe.

```
procedure  GrafoaKorritu (G: in Grafoa) is
   Aztertua: constant Boolean := True;       // n= # erpinak; a= # arkuak
   Bisitak: Taula(1..n):= (Others=>  not (Aztertua));
begin
   for  Erp  in  1..n  loop
         if  not MarkatuaDago(Bisitak, Erp)  then
               MarkaIpini(Bisitak, Erp, Aztertua);
               SK(G, Erpina);   // ZK SK(G, Erp);
         end if;
   end loop;
end GrafoaKorritu;


procedure  SK  (G: in Grafoa; A: in Erpina) is
   AAuzokideenKopia: ErpinenL;  Lehena: Erpina;
begin
   AAuzokideen_Kopia:= Auzokideak(G, A);
   while  not( HutsaDaL? (AAuzokideenKopia))  loop
      Lehena:= LehenengoaL(AAuzokideenKopia);
      Hondarra(A_AuzokideenKopia);
      if not Markatua_Dago(Bisitak, Lehena)  then
         Marka_Ipini(Bisitak, Lehena, Aztertua);
         SK(G, Lehena);
      end if;
   end loop;
end SK;


procedure  ZK  (G: in Grafoa; X: in Erpina) is
   Ilara: IlaraDatuMota:=HutsaI;  YRenAuzokideak: ErpinenL;
   Y,Z: Erpina;
begin
   IlarariGehitu(Ilara, X);
   while not( HutsikDagoI?(Ilara))  loop
      Y:= Burua(Ilara);  Ilara:= IlaratikKendu(Ilara);
      YEnAuzokideak:= Auzokideak(Y).;
      while  not ( HutsikDagoL?(YRenAuzokideak))  loop
         Z:= LehenaL(YRenAuzokideak);  Hondarra(YRenAuzokideak);
         if  not MarkatuaDago?(Bisitak, Z)  then
               MarkaIpini(Bisitak, Z, Aztertua);
               IlarariGehitu(Ilara, Z);
         end if;
        end loop;
   end loop;
end ZK;
```

UPV/EHU—Informatika Ingenieritza
DEA-II 1 PARTZIALA (2011-iii-8) - E taldea
SOLUZIO BAT

1. (2 pt) *Egiazkoa* edo *faltsua* balioekin osatu ondorengo taulako gelaxkak, behar den bezala, azkeneko zutabean kostu asintotiko txikien duen funtzioa adieraziz:

f(n)	g(n)	f(n)∈O(g(n))	f(n)∈Ω(g(n))	f(n)∈Θ(g(n))	Onena
$\sqrt{n} * (\lg n)$	n	*True*	*False*	*False*	$\sqrt{n} * (\lg n)$
$4^{\log_2 n}$	$2n^2$	*True*	*True*	*True*	$2n^2$
$n\sqrt{n}$	$3n^{1.4}$	*False*	*True*	*False*	$3n^{1.4}$

2. (2 pt) A(i..j) bektoreko k balio txikiena *KHautaketa*(A,i,j,k) algoritmoak kalkulatzen duela ikusia dugu. Aplika ezazu *KHautaketa* k=7 eta ondorengo bektoreari:

A= « 46, 27, 2, 75, 83, 9, 51, 6, 58, 55 »

Deien ondorioz suertatuko balioekin osatu tauletako gelaxkak. Gainera, deien burukoen balio konkretuak zehaztea ere eskatzen zaizu (hots, azpitaularen hasiera eta bukaera indizeen balioak hala nola zenbat garren balio txikiena bilatzen den).

KHautaketa (A, 1, 10, 7) *Banaketa*(A, *1, 10*, BP) // non BP= *5* }

6	27	2	9	46	75	51	83	58	55

KHautaketa (A, *6, 10, 7*) *Banaketa*(A, *6, 10*, BP) // non BP= 9 }

azpibektorea pasa izan bagenio: 2. elementua

					55	51	58	75	83

KHautaketa (A, *6, 8, 7*) *Banaketa*(A, 6 , 8 , BP) // non BP= *7* }

azpibektorea pasa izan bagenio: 2. elementua

					51	55	58		

3. (2 pt) Ondorengo algoritmoaren denbora ordena kalkulatu.

```
proz AzterNazazu (M) is
(1) MetaEraiki(M);
(2) b ← (M(1) + M(M.last))/2;
(3) N ← M.last;
(4) for  i  in  1..N  loop
(5)   b ← b + (-1)^i M(i);
(6)   M.last← M.last+1;  M(M.last)←b;  Azaleratu(M, M.last);
(7)   b ←M(1);  M(1)←M(M.last);  M.last←M.last-1;  Hondoratu(M,1);
(8) end loop;
end proz.
```

(1) MetaEraiki lineala da osagai kopuruan: Θ(n)
(2) eta (3) zenbait eragiketa: Θ(1)
(4) for egitura N aldiz errepikatzen da:
   (5) T. ktea behar duen eragiketa
   (6) bi eragiketa T. ktea behar dutena eta Azaleratu azkeneko elementua (erroraino kasu txarrenean eta hortaz) logaritmikoa dena osagai kopuruan, O(lg n).
   (7) bi eragiketa T. ktea behar dutena eta Hondoratu erroa (kasu txarrenean hosto izateraino, hortaz, logaritmikoa dena osagai kopuruan, , O(lg n)..

(5), (6) eta (7) eragiketak: Θ(1)+ O(lg n)+O (lg n) =$_{\text{baturaren erregela aplikatuz}}$= O(lg n)
(4) For-aren kostua: O(n lg n)
Lau agindu nagusien sekuentziaren denbora kostuari baturaren erregela aplikatuz:
O(n lgn) da AzterNazazu prozeduraren **denbora ordena**

4. (2 pt) Zenbaki osokoen bektore bat emanik, osagai handiena eta txikiena kalkulatzen duen eta *2n-3* konparazio egiten dituen soluzioa bilatzea erraza da. Bestalde, badakigu problema hori ebazten duten algoritmoek Θ(n) denbora ordena dutela. Oraingoan, *zatitu eta irabazi teknika* erabiliz eta *2n-3* baino konparazio gutxiago burutzen dituen soluzio bat gara dezazula eskatzen dizugu. Zure algoritmoak egiten dituen konparazio kopuru zehatza ere kalkula ezazu.

Sol:

```
procedure MIN_MAX  ( B: in BEKTOREA; MIN, MAX: out INTEGER) is
 MIN1,MIN2,MAX1,MAX2: INTEGER;

begin
   if B'LENGTH=1 then MIN= then MAX:= B(B'FIRST); MIN:= B(B'FIRST);
 else if B'LENGTH=2 then
if B(B'FIRST)> B(B'LAST) then MAX:= B(B'FIRST); MIN:= B(B'LAST);
       else MAX:= B(B'LAST); MIN:= B(B'FIRST);
       end if;
 esle  MIN_MAX(B(B'FIRST..(B'FIRST+B'LAST)/2),  MIN1, MAX1);
       MIN_MAX(B((B'FIRST+B'LAST)/2)+1)..B'LAST),  MIN2, MAX2);
       if MAX1 > MAX2  then  MAX:= MAX1;
       else MAX:= MAX2;
       end if;
       if MIN1 < MIN2  then  MIN:= MIN1;
       else MIN:= MIN2;
       end if;
 end if;
end  MIN_MAX ;
```

Algoritmoa honek egiten dituen konparazio kopurua honako ekuazioak garatuz lortuko dugu:

$T(1)=0;$
$T(2)=1$
$T(n) = 2\,T(n/2) + 2 =$
$= 2(2\,T(n/2^2) + 2)+2= 2^2\,T(n/2^2)+2^2 + 2$
$= 2^i\,T(2^i)+2^i + 2$
$= 2^i\,T(2)+2^i + 2= 2^i\,1+ 2^i\,2+ 2= n/2+ 2*n/2 - 2=$ **3/2 n -2**

5. (2 pt) Grafo zuzendu batean, *jatorri* eta *jomuga* erpinak seinalatu ditugu. Gainera erpin *Debekatu*en multzo bat ere badugu. Egizu algoritmo bat erabaki dezan jatorritik jomugara bidea dagoen ala ez, grafoko arkuak zeharkatuz baina erpin Debekatuak zeharkatu gabe.

SOL:

```
procedure  GrafoaKorritu (G: in Grafoa; jatorri,jomuga: in Erpinak;
                          Debekatuak: in Taula(1..n) of Boolean;
                          Konektatuak: out Boolena) is
   Aztertua: constant Boolean := True;      // n= # erpinak; a= # arkuak
   Bisitak: Taula(1..n):= (Others=>  not (Aztertua));
   Bidea: Taula(1..n) of Boolean;
begin
   MarkaIpini(Bisitak, jatorria, Aztertua);
   Bidea(jatorria):=False;
   SK(G, jatorria, jomuga);
   Konektatuak := Bidea(jatorria);
   end loop;
end GrafoaKorritu;


procedure  SK  (G: in Grafoa; A, jomuga: in Erpina) is
   AAuzokideenKopia: ErpinenL;   Lehena: Erpina;
begin
   AAuzokideen_Kopia:= Auzokideak(G, A);
   while  not( HutsaDaL? (AAuzokideenKopia)) and not (Bidea(A))  loop
      Lehena:= LehenengoaL(AAuzokideenKopia);
      Hondarra(A_AuzokideenKopia);
      if (Lehena=jomuga) then Bidea(A):=True;
      else     if not Markatua_Dago(Bisitak, Lehena)
                  and not (Debekatua(Lehena))
               then  Bidea(Lehena):=False;
                     Marka_Ipini(Bisitak, Lehena, Aztertua);
                     SK(G, Lehena, jomuga);
               end if;
               Bidea(A):= Bidea(A) OR Bidea(Lehena);
   end loop;
end SK;
```