Osoko_Bektore datu-mota (IV.GAIA)

Helburuak:Osoko_Bektore eta Karaktere_Bektore datu-motak erabiltzen ikastea. Osoko_Bektore eta Karaktere_Bektore hainbat balio duen array-ak dira.

```
N : constant Integer := 10;

type Osoko_Bektore is array (1 .. N) of Integer;

type Karaktere_Bektore is array (1 .. N) of Character;
```

1. Bektorearen batezbesteko aritmetikoa

B osoko-bektore bat emanda, bektoreko zenbakien batezbesteko aritmetikoa kalkulatzen duen algoritmoa espezifikatu eta egin. Azpiprograma modura inplementatu.

```
funtzio Batezbestekoa (B: datu Osoko_Bektore)
                  itzuli  Erreala
--Aurre: N osokoa B bektoreko indize posible bat da.
--Post:  Itzuli da zenbaki erreal bat, B bektoreko
-- osagaien batezbesteko aritmetikoa adierazten duena
hasiera
   Batura := 0
   egin I guztietarako B'First tik B'Last raino
         Batura := Batura + B(I)
   amguztietarako
   itzuli Erreal_Bihurtu (Batura)
          / Erreal_Bihurtu (B'Length)
amaia
```

2. Bektorea idatzi

B osoko-bektore bateko elementuak inprimatuko dituen algoritmoa espezifikatu eta egin. Azpiprograma modura inplementatu.

```
procedure Bektorea_Idatzi (B: in Osoko_Bektorea) is
-- Postbaldintza: Sek (Osokoen Sekuentzia)
                  non B bektorearen osagaiak dauden.
```

```
begin
   for I in B'First .. B'Last loop
      Idatzi_Osokoa (B(I)) ;
   end loop;
end Taula_Idatzi;
```

3. ‘A’ karakterea zenbat aldiz karaktere-bektorean

B karaktere-bektore bat emanda, 'A' letra zenbat aldiz agertzen den kontatu. Azpiprograma modura inplementatu.

```
funtzio A_Letrak_Kontatu (B: datu Karaktere_Bektore)
                        itzuli  Osokoa
Post: Itzuli da zenbaki oso bat, B bektoreko
   osagaien artean ‘A’ letraren agerpen-kopurua
hasiera
   Kont := 0
   egin I guztietarako B'First tik B'Last raino
      baldin B(I)= 'A' orduan
         Kont := Kont + 1
      ambaldin
   amguztietarako
   itzuli  Kont
amaia
```

4. Zenbat bokal karaktere-bektorean

KB karaktere-bektore bat emanda, karaktere horien guztien artean zenbat bokal agertzen diren kontatu.

```
funtzio Bokalak_Kontatu (B: datu Karaktere_Bektore)
                       itzuli  Osokoa
Post:  Itzuli da zenbaki erreal bat, B bektoreko
   osagaien artean zenbat bokal dauden adierazten duena
hasiera
   Kont := 0
   egin  I guztietarako B'First tik B'Last raino
      baldin B(I) = 'A'  edo B(I) = 'E'  edo B(I) = 'I'
             edo  B(I) = 'O'  edo B(I) = 'U'
      orduan
         Kont := Kont + 1
      ambaldin
   amguztietarako
   itzuli  Kont
amaia
```

5. Zenbat ez-bokal karaktere-bektorean

KB karaktere-bektore bat emanda, karaktere horien guztien artean bokalak ez direnak kontatu. Azpiprograma modura inplementatu. Karaktere bat bokala den ala ez aztertzen duen funtzio bat definitu eta erabili.

```
funtzio Bokala_izan (Kar : Karakterea) itzuli  Boolearra
Postbaldintza: egiazkoa itzuliko da, karakterea bokala
bada; faltsua bestela
hasiera
 itzuli (B(I) = 'A'  edo B(I) = 'E'  edo B(I) = 'I' edo
         B(I) = 'O'  edo B(I) = 'U' )
amaia

funtzio Ez_Bokalak_Kontatu (B: datu Karaktere_Bektore)
                        itzuli  Osokoa
Aurrebaldintza: N>0
Postbaldintza: Itzuliko da zenbaki erreal bat, B
bektoreko osagaien artean bokalak ez diren karaktereak
zenbat diren adierazten duena
hasiera
   Kont := 0
   egin  I guztietarako B'First tik B'Last raino
      baldin ez Bokala_Izan (B(I))
      orduan
         Kont := Kont + 1
      ambaldin
   amguztietarako
   itzuli  Kont
amaia
```

6. Zenbat aldiz errepikatzen den "TA" karaktere-bikotea

KB karaktere-bektore batean 'A' karakterea 'T' karaktere baten atzetik zenbat aldiz agertzen den. kontatu Azpiprograma modura inplementatu.

```
 funtzio TA_Kontatu (B: datu Karaktere_Bektore)
                    itzuli Osokoa
Postbaldintza: Itzuliko da zenbaki oso bat, B bektoreko
osagaien artean zenbat aldiz agertzen den 'A' karakterea
'T' karaktere baten atzetik adierazten duena
hasiera
   Kont := 0
   egin  I guztietarako B'First tik B'Last-1 raino
      baldin  B(I) = 'T'  eta  B(I+1) = 'A'
      orduan
         Kont := Kont + 1
      ambaldin
  amguztietarako
  itzuli  Kont;
amaia
```

7. Zenbat zenbaki lehen osoko-bektorean

B osoko-bektore bat emanda, zenbaki lehenen kopurua kalkulatuko duen algoritmoa espezifikatu eta egin. Azpiprograma modura inplementatu.

```
funtzio Zenbat_Lehenak (B: datu Osoko_Bektore)
                  itzuli  Osokoa
Postbaldintza: Itzuliko da zenbaki erreal bat, B
bektoreko osagaien artean zenbat diren lehenak
hasiera
   Kont := 0
   egin  I guztietarako B'First tik B'Last raino
      baldin Lehena_Da (B(I))
      orduan
         Kont := Kont + 1
      ambaldin
   amguztietarako
   itzuli  Kont
amaia
```

8. Bektoreko elementu maximoa eta bere posizioa

B osoko-bektore batean zenbaki maximoa eta bere posizioa bilatzeko algoritmoa espezifikatu eta egin. Azpiprograma modura inplementatu.

```
algoritmoa Bilatu_Maximoa (B : datu Osoko_Bektore ;
                           Maximoa, Pos : emaitza Osokoa)
Post: Maximoa delakoak B bektoreko osagaien arteko
  maximoa adierazten du.
   Pos delakoak B bektoreko osagaien arteko maximoaren
   posizioa adierazten du.
hasiera
   Maximoa := B(B'First)
   egin I guztietarako B'First tik B'Last raino
      baldin B(I)> Maximoa  orduan
         Maximoa := B(I)
         Pos := I
      ambaldin
   amguztietarako
amaia
```

9. Zenbakia bilatu osoko-bektore ez-ordenatuan

B osoko-bektore ez-ordenatu batean zenbaki bat bilatzeko algoritmoa espezifikatu eta egin. Zenbakia bektorean badago, lehenengo agerpenaren posizioa itzuli beharko da; eta bestela, ez dagoenean, zero itzuli beharko da. Azpiprograma modura inplementatu.

```
funtzio Bilatu_Posizioa (B : datu Osoko_Bektore ;
                         X : datu Osokoa)
                      itzuli  Osokoa
Aurrebaldintza:
Postbaldintza: Itzuliko da osoko bat, X zenbakia B bektoreko
   zenbatgarren posizioan dagoen adierazten duena;
   X zenbakia B bektoreko osagaien artean ez badago,
     0 itzuliko da
hasiera
   I := B'First
   bitartean  I< B'Last eta X /= B(I) egin
      I := I + 1
   ambitartean
   -- I=B'Last edo X=B(I)
   baldin X=B(I) orduan
      itzuli  I
   bestela  itzuli 0
   ambaldin
amaia
```

kontuz ibiliz gertatzen da honela definituz gero?

```
hasiera
   I := B'First
   bitartean  I<=B'Last eta X /= B(I) egin
      I := I + 1
   ambitartean
   baldin I<=B'Last orduan
      itzuli  I
   bestela  itzuli 0
   ambaldin
amaia
```

Errorea gerta liteke zenbakia ez badago bektorean, B(B'Last+1) ebaluatu nahi izango baita denean horrelakorik existitzen ez bada. Beste aukera bat:

```
funtzio Bilatu_Posizioa (B : datu Osoko_Bektore ;
                         X : datu Osokoa) itzuli Osokoa
Aurrebaldintza:
Postbaldintza: Itzuliko da osoko bat, X zenbakia B
   bektoreko zenbatgarren posizioan dagoen;
   X zenbakia B bektoreko osagaien artean ez badago,
   0 itzuliko da
```

```
hasiera
   Pos := 0
   I := B'First
   bitartean  I<=B'Last  eta  Pos/=0 egin
      baldin X=B(I) orduan
         Pos := I
      else
         I := I + 1
      ambaldin
   ambitartean
   -- I=B'Last+1 edo X=B(I)
   itzuli Pos
amaia
```

10. Zenbakia bilatu osoko-bektore ordenatuan

B osoko-bektore ordenatu batean (txikienetik handienera) zenbaki bat bilatzeko algoritmoa espezifikatu eta egin. Zenbakia bektorean badago, posizioa itzuli beharko da; eta bestela, ez dagoenean, zer posiziotan kokatu beharko genukeen eman. Zenbakia bektoreko guztiak baino handiagoa bada, B'Last+1 itzuli beharko da. B'Lengthzero bada, 1 itzuli beharko da. Azpiprograma modura inplementatu.

Aurreko ariketako soluzio bera erabil daiteke, baina kasu honetan elementua bektorean ez dag oela jakin daiteke, bera baino handiago den zenbaki bat aurkitzen dugunean. Beraz, begiztako baldintza aldatu beharko da, kasu horietan bilaketa askoz lehenago bukatzeko.

Lehen :

```
bitartean  I< B'Last eta X /= B(I) egin
```

Orain :

```
bitartean  I< B'Last eta X < B(I) egin
```

B'Length=0 kasua bereiztu egin beharko da.

```
funtzio Bilatu_Posizioa_Ord (B: Osoko_Bektore;
                             N, X: Osokoa)
   itzuli  Osokoa
Aurrebaldintza:
   B bektoreko elementuak ordenatuta daude
   txikienetik handienera
Postbaldintza: Itzuliko da osoko bat,
   B bektoreko zenbatgarren posiziotan kokatu beharko
   litzateke adierazten duena.
   Zenbakia bektoreko guztiak baino handiagoa bada,
   B'Last+1 itzuli beharko da
```

```
I :Osokoa
hasiera
   baldin B'Length=0 orduan
      itzuli 1
   bestela
      I := 1
      bitartean  I< B'Last eta X >B(I) egin
         I := I + 1
      ambitartean -- I=B'Last   edo   X<= B (I)
      baldin X<=B(I) orduan
         itzuli  I
      bestela  -- I=B'Last eta X>B(I)
         itzuli B'Last+1
      ambaldin
   ambaldin
amaia
```

11. Posizio bat eskuinera mugitu

B osoko-bektore bat emanda, [I1,I2] tarteko osagai guztiak posizio bat eskuinera mugitzen dituen algoritmoa espezifikatu eta egin. Azpiprograma modura inplementatu.

```
procedure Mugitu1 (B1: in out Osoko_Bektorea;
                   I1,I2: in Integer ) is
-- Aurrebaldintza: B1'First<=I1<=I2<=B'Last-1
-- Postbaldintza: B1 bektoreko [I1,I2] tarteko osagaiak
--                posizio bat eskuinera mugitu dira.
begin
   for N in reverse I1..I2 loop
      B1(N+1) := B1(N);
   end loop;
end Mugitu1;
```