Javított buborékrendezés II.

```
eljárás javítottbuborékrendezés2 (A : tömbtípus);
  változó i, j, cserevám : egész;
  változó csere : elemtípus;
  i := n;
  amíg i >= 2 ismétel
    cserevám := 0;
    ciklus j := 1 .. i-1 ismétel
      ha A[j] > A[j+1] akkor
        csere := A[j];
        A[j] := A[j+1];
        A[j+1] := csere;
        cserevám := j;
      h vége;
    c vége;
    i := cserevám;
  c vége;
vége;
```

Beszúró rendezés:

```
eljárás beszúrórend (A: tömbtípus);
  változó j, i: egész;
  változó x: elemtípus;
  ciklus i:= (2..n)
    j:= i-1;
    x:= A[i];
    amíg (j>=1) és (x<A[j])
      A[j+1]:= A[j];
      j:= j-1;
    a vége;
    A[j+1]:= x;
  c vége;
vége;
```

hatékonyság:

helyfoglalás : $n+1$

összehasonlítások száma: rendezett sorozat esetén : $n-1$

fordított rendezettségnél : $n(n-1)/2$

értékadások száma: rendezett sorozat esetén : $2 \cdot (n-1)$

fordított rendezettségnél : $2 \cdot (n-1) + 3 \cdot n \cdot (n-1)/2$

Shell rendezés:

```
eljárás shellberzúrással ( A : tömbtípus);
  változó i, j, d, c : egész;
  változó x : elemtípus;
  d := n;
  ismételd
    d := d/3 + 1;
    ciklus c := (1..d) ismételd
      j := c + d;
      amíg j <= n ismételd
        i := j - d;
        x := A[j];
        amíg (i >= 1) és (x < A[i]) ismételd
          A[i+d] := A[i];
          i := i - d;
        a vége;
        A[i+d] := x;
        j := j + d;
      a vége;
    c vége;
  i vége d = 1 esetén;
vége;
```

Összefuttatás:

```
eljárás összefuttatás (A: tömbtípus1; B: tömbtípus2; C: tömbtípus3);
  változó i,j,k: egész;
  k:=0;
  i:=1;
  j:=1;
  amíg (i<=n) és (j<=m) ismétel
    k:=k+1;
    elágazás
      amikor A[i]<B[j]:
        C[k]:=A[i]; i:=i+1;
      amikor A[i]>B[j]:
        C[k]:=B[j]; j:=j+1;
      különben
        C[k]:=A[i]; i:=i+1; j:=j+1;
    e vége;
  a vége;
  amíg i<=n ismétel
    k:=k+1;
    C[k]:=A[i]; i:=i+1;
  a vége;
  amíg j<=m ismétel
    k:=k+1;
    C[k]:=B[j]; j:=j+1;
  a vége;
vége;
```

VEREM:

deklarálás:

```
konstans  maxelem = maximális_elemszám ;
típus     elemtípus : tárolandó_elemek_típusa
típus     veremtípus : rekord (
              elem : tömb [1.. maxelem] elemtípus
              alja, teteje : egész
          )
```

inicializálás:

```
eljárás veremkezd (verem : veremtípus);
  verem . alja := 1;
  verem . teteje := Ø;
vége;
```

verem állapotának lekérdezése:

```
függvény veremüres (verem : veremtípus) : logikai;
    veremüres := (verem . teteje < verem . alja)
vége;

függvény veremtele (verem : veremtípus): logikai ;
    veremtele := (verem . teteje = maxelem);
vége;
```

Új elem elhelyezése a veremben:

```
függvény verembe (verem : veremtípus; Adat : elemtípus) : logikai;
  ha nem veremtele (verem) akkor
    verem.teteje := verem.teteje + 1;
    verem.elem[verem.teteje] := Adat;
    verembe := igaz
  különben
    verembe := hamis;
  ha vége;
vége;
```

Elem kivétele a veremből:

```
függvény veremből (verem : veremtípus; Adat : elemtípus) : logikai;
  ha nem veremüres (verem) akkor
    Adat := verem.elem[verem.teteje];
    verem.teteje := verem.teteje - 1;
    veremből := igaz
  különben
    veremből := hamis;
  ha vége;
vége;
```

SOR:

deklarálás:

```
konstans    maxelem = maximális_elemszám
típus       elemtípus : társadó_elemek_típusa
típus       sortípus : record (
                elem : tömb [ 1 .. maxelem ] elemtípus
                első, utolsó : egész
            )
```

inicializálás:

```
- eljárás sorrend ( sor : sortípus );
  sor . első := 1;
  sor . utolsó := Ø;
vége;
```

sor állapotának lekérdezése:

```
függvény sorüres ( sor : sortípus ) : logikai ;
  sorüres := ( sor.első > sor . utolsó);
vége;

függvény sortele ( sor : sortípus) : logikai;
  sortele := ( sor . utolsó = maxelem ) ;
vége;
```

Új elem elhelyezése a sorban:

```
függvény sorba (sor: sortípus; Adat: elemtípus): logikai;
  ha nem sortele (sor) akkor
    sor.utolsó := sor.utolsó + 1;
    sor.elem[sor.utolsó] := Adat;
    sorba := igaz
  különben
    sorba := hamis;
  ha vége;
vége;
```

Elem kivétele a sorból:

```
függvény sorból (sor: sortípus; Adat: elemtípus): logikai;
  ha nem sorüres (sor) akkor
    Adat := sor.elem[sor.első]
    sor.első := sor.első + 1;
    sorból := igaz
  különben
    sorból := hamis;
  ha vége;
vége;
```