TÖL304G
Forritunarmál
Verkefnablað 8

Snorri Agnarsson

13. október 2019

Skilatími

Í öllum verkefnunum skulu öll föll, þar með talið hjálparföll, hafa skýra og rétta lýsingu með „Notkun: ...“, „Fyrir: ...“ og „Gildi: ...“. Takið eftir að í sumum tilfellum þurfa forskilyrði að innihalda lýsingar á sama sniði fyrir viðföng sem eru föll, og svipað gildir í eftirskilyrði (þ.e. „Gildi: ...“) fyrir gildi sem eru föll.

Athugið að í fjölnota einingum, svo sem forgangsbiðraðaeiningum eins og hér að neðan, þurfa innflutt föll og aðgerðir einnig að hafa slíka lýsingu.

Verkefni

Hópverkefni

Skilið eftirfarandi:

1. Hönnunarskjali fyrir fjölnota forgangsbiðröð í Morpho. Útfluttar aðgerðir (stef) skulu meðal annars bjóða upp á eftirfarandi:
   - Að smíða nýja tóma forgangsbiðröð. Þetta gæti verið fall sem tekur ekkert viðfang og skilar forgangsbiðröð af ótakmarkaðri stærð. Þetta gæti líka verið fall sem tekur heiltöluviðfang sem væri hámarksstærð forgangsbiðraðarinnar.
   - Að athuga hvort gefin forgangsbiðröð er tóm.
   - Að athuga hvort gefin forgangsbiðröð er full. Þessi aðgerð er ekki nauðsynleg ef allar forgangsbiðraðir eru af ótakmarkaðri stærð.

- Að bæta tilteknu gildi inn í tiltekna forgangsbiðröð með tilteknum forgangi (lykli). Þetta skal vera aðgerð sem tekur tvö viðföng, annað viðfangið er gildið, hitt er lykillinn.
- Að fjarlægja og skila *fremsta* gildi úr tiltekinni forgangsbiðröð (þetta gætu verið tvær eða fleiri mismunandi aðgerðir). Fremsta gildið er það gildi sem hefur lykil með mesta forgang.

Forgangsbiðröðin skal vera útfærð sem ein eining í Morpho og skal samanburðaraðgerð fyrir lykla (forganga) vera innflutt (ótengd). Einingin skal vera *fjölnota* þannig að tengja megi mismunandi samanburðaraðgerðir til að forgangsraða gildum með mismunandi gerðum lykla svo sem heiltölum, fleytitölum og strengjum, svo eitthvað sé nefnt.

2. Útfærslu (smíð, forritstexti) fyrir forgangsbiðröðina sem lýst er í hönnunarskjalinu. Smíðin skal hafa innri skjölun í formi fastayrðingar gagna, sem lýsir því hvernig forgangsbiðraðir eru geymdar í minni tölvunnar.

3. Tvö prófunarforrit (forritstexti) fyrir forgangsbiðraðirnar. Forritin skulu nota forgangsbiðröð til að raða tölum, annars vegar í vaxandi röð og hins vegar í minnkandi röð.

   Forritin skulu skrifa runu talna í vaxandi eða minnkandi röð, eftir atvikum, og sýna skal útkomur prófana.

   Athugið að það er ekkert því til fyrirstöðu að lykill og samsvarandi gildi séu það sama í forgangsbiðröðinni. Það er þægileg aðferð þegar við notum forgangsbiðröð til að raða.

Einstaklingsverkefni

Skrifið Morpho einingu fyrir hlaða (*stack*) gilda af hvaða tagi sem er. Skila skal eftirfarandi:

1. Hönnunarskjal sem inniheldur lýsingar á öllum útfluttum og innfluttum (ef einhverjar eru) aðgerðum (föllum) í einingunni. Lýsingarnar felast í liðunum Notkun:..., Fyrir:... og Eftir:....

2. Fastayrðing gagna fyrir hlaðana.

3. Forritstexti fyrir einingunа.

4. Forritstexti fyrir prófunarforrit.

5. Úttak úr prófunarforriti.

Innsetning gildis í lista

Eftirfarandi getur verið gagnlegt fyrir forgangsbiðröðina. Þennan forritstexta má finna í Uglunni (`insert.morpho`). Athugið að forgangsbiðröð sem notar fall líkt þessu sem sinn kjarna verður varla hraðvirk, en ekki er beðið um hraðvirka forgangsbiðröð. Hraðvirka forgangsbiðröð mætti útfæra með því að geyma pörin (lykill,gildi) í hrúgu (heap), en það krefst þess að annaðhvort sé hver forgangsbiðröð af fastri hámarksstærð eða að fylkinu sem skipað er í hrúgu sé endurúthlutað þegar minnisrými þess þrýtur.

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; Hönnunarskjal fyrir eininguna "insert" ;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Innflutt (Imported):
;;;  Notkun: z = x <<== y
;;;  Fyrir:  x og y eru gildi af þeirri gerð sem við
;;;          viljum bera saman
;;;  Eftir:  z er satt ef x verður að vera á undan y,
;;;          ósatt ef y verður að vera á undan x.
;;;  Ath.:   Skilagildið má vera hvað sem er ef x má
;;;          vera á undan y OG y má vera á undan x.
;;;
;;; Útflutt (Exported):
;;;  Notkun: y = insert(x,u)
;;;  Fyrir:  x er listi gilda af þeirri gerð sem <<==
;;;          ræður við, í vaxandi röð miðað við <<==,
;;;          u er einnig gildi af þeirri gerð.
;;;  Eftir:  y er listi í vaxandi röð miðað við <<==
;;;          sem inniheldur öll gildin úr x auk u.
;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; Hönnunarskjali lýkur hér ;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

"insert.mmod" =
!
{{
insert =
    fun(x,u)
    {
        x==[] && (return [u]);
        head(x) <<== u || (return u:x);
        head(x) : insert(tail(x),u);
    };
}}
;

show "insert.mmod";

"test1.mexe" = main in
```

```
{{
main =
    fun()
    {
        var x=[1,9,2,8,3,7,4,6,5],y;
        while( x!=[] )
        {
            y = insert(y,head(x));
            x = tail(x);
        };
        writeln(y);
    };
}}
*
"insert.mmod"
*
{{
<<== = fun <=(x,y);
}}
*
BASIS
;

"test2.mexe" = main in
{{
main =
    fun()
    {
        var x=[1,9,2,8,3,7,4,6,5],y;
        while( x!=[] )
        {
            y = insert(y,head(x));
            x = tail(x);
        };
        writeln(y);
    };
}}
*
"insert.mmod"
*
{{
<<== = fun >(x,y);
}}
*
BASIS
;
```

Fjarlæging stærsta gildis úr lista

Eftirfarandi getur einnig verið gagnlegt fyrir forgangsbiðröðina. Þennan forritstexta má einnig finna í Uglunni (`removemax.morpho`). Forgangsbiðröð sem notar þetta fall verður varla hraðvirk.

```
{;;;

Hönnunarskjal

Útfluttar aðgerðir
==================

Notkun: y = removeMax(x);
Fyrir:  x er listi, ekki tómur, sem inniheldur gildi
        sem bera má saman með innfluttu aðgerðinni >.
Eftir:  y er listi sem inniheldur öll gildin úr x nema
        það gildi sem er stærst miðað við >.  Það er
        þá ekkert annað stak z í listanum x þannig að
        z>y er satt (nema, reyndar megum við leyfa að
        z>y ef z og y eru sama gildi).

Innfluttar aðgerðir
===================

Notkun: b = x>y;
Fyrir:  x og y eru gildi af því tagi sem við viljum bera
        saman, þ.e. gildi af því tagi sem mega vera í
        inntakslistanum x í útfluttu removeMax aðgerðinni.
Eftir:  b er satt ef x er stærra en y, annars ósatt.
Athugasemd: Þessi aðgerð verður að virka sem samanburðaraðgerð,
        þ.e. verður að vera gegnvirk og svo framvegis. Sérhver
        tvö gildi þurfa að vera samanburðarhæf þannig að annað
        gildið sé stærra en hitt, nema gildin séu jöfn.  Bæði
        má nota samanburðaraðgerðir sem virka eins og < og >,
        eða samanburðaraðgerðir sem virka eins og <= og >=.
        Samanburðaraðgerðin má því vera andsamhverf (eins og >=)
        eða ósamhverf (eins og >).

;;;}

"removemax.mmod" =
{{
removeMax =
    fun(x)
    {
        var prev=null, currMax=head(x), z=x;
        while( tail(z) )
        {
            ;;; z vísar á einhvern hala á listanum x,
```

```
            ;;; eins og hér er sýnt:
            ;;;
            ;;;  x=[x1,x2,...,xI,...,xN]
            ;;;            z=[xI,...,xN]
            ;;;
            ;;; currMax er stærsta gildið af x1,...,xI.
            ;;; Ef currMax er x1 þá er prev==null, annars
            ;;; vísar prev á næsta hlekk fyrir framan þann
            ;;; sem inniheldur currMax.
            ;;;
            if( head(tail(z))>currMax )
            {
                prev = z;
                currMax = head(tail(z));
            };
            z = tail(z);
        };
        if( !prev ) { return tail(x) };
        setTail(prev,tail(tail(prev)));
        x;
    };
}}
;

"test1.mexe" = main in
!
{{

;;; Notkun: y = max(x);
;;; Fyrir:  x=[x1,x2,...,xN] er listi
;;;         gilda sem samanburðaraðgerðin >
;;;         virkar fyrir.
;;; Eftir:  y er stærsta gildið í x miðað
;;;         við samanburðaraðgerðina >.
max =
    fun(x)
    {
        var mx = head(x);
        var z = x;
        for(;;)
        {
            ;;; z vísar á einhvern hala á listanum x,
            ;;; eins og hér er sýnt:
            ;;;
            ;;;  x=[x1,x2,...,xI,...,xN]
            ;;;            z=[xI,...,xN]
            ;;;
            ;;; mx er stærsta gildið af x1,...,xI.
            ;;;
            z = tail(z);
```

```
95              if( !z ) { return mx };
96              if( head(z)>mx ) { mx=head(z) };
97          };
98      };
99
100 ;;; Notkun: main();
101 ;;; Fyrir:  Ekkert.
102 ;;; Eftir:  Búið er að skrifa út tölurnar
103 ;;;         9, 8, 7, 6, 5, 4, 3, 2, 1, eina
104 ;;;         í hverri línu.
105 main =
106     fun()
107     {
108         var x = [1,9,2,8,3,7,4,6,5];
109         while( x )
110         {
111             writeln(max(x));
112             x = removeMax(x);
113         };
114     };
115 }}
116 *
117 "removemax.mmod"
118 *
119 BASIS
120 ;
```

Þessi tvö föll geta, sem sagt, verið gagnleg í einfaldri útfærslu forgangsbiðraða. Hvert um sig er gagnlegt fyrir tiltekna fastayrðingu gagna, sem rætt verður í fyrirlestri. Hér er um tvær mismunandi fastayrðingar gagna að ræða þannig að aðeins annað fallanna verður gagnlegt.