Tölvunarfræði 2/2a
Svör við skyndiprófi

Snorri Agnarsson

27. febrúar 2008

1. **Spurning:** Fyllið inn þar sem spurningarmerkin eru, þ.e. skrifið hvað á að koma í stað `?1?`, o.s.frv. Þetta eru samtals átta svör.

```
// Notkun: k = leita(f,i,j,x);
// Fyrir:  f[i..j-1] er í vaxandi röð
// Eftir:  f[i..j-1] er óbreytt, i <= k <= j, og
//         f[i..k-1] < x <= f[k..j-1]
int leita( double[] f, int i, int j, double x ) {
  if( i==j ) return i;
  int m = (i+j)/2;
  if( f[m] ?1? x )
    return leita(f,i,?2?,x);
  else
    return leita(f,?3?,j,x);
}

int leita( double[] f, int i, int j, double x ) {
  int p=i, q=j;
  while( ?4? ) {
    // | <x | óþekkt | >=x |
    //  i    p        q     j
    int m = (?5?)/2;
    if( f[m] ?6? x )
      p = ?7?;
    else
      q = ?8?;
  }
  return p;
}
```

Svar:

```
// Notkun: k = leita(f,i,j,x);
```

```
// Fyrir:  f[i..j-1] er í vaxandi röð
// Eftir:  f[i..j-1] er óbreytt, i <= k <= j, og
//         f[i..k-1] < x <= f[k..j-1]
int leita( double[] f, int i, int j, double x ) {
  if( i==j ) return i;
  int m = (i+j)/2;
  if( f[m] >= x )
    return leita(f,i,m,x);
  else
    return leita(f,m+1,j,x);
}

int leita( double[] f, int i, int j, double x ) {
  int p=i, q=j;
  while( p != q ) {
    // | <x | óþekkt | >=x |
    //  i    p        q     j
    int m = (p+q)/2;
    if( f[m] < x )
      p = m+1;
    else
      q = m;
  }
  return p;
}
```

2. **Spurning:** Fyllið inn þar sem spurningarmerkin eru.

```
// Notkun: swap(f,i,j);
// Fyrir:  f[i] og f[j] eru sæti í f
// Eftir:  Búið er að víxla gildunum í f[i] og
//         f[j]
void swap( double[] f, int i, int j );

// Notkun: sort(f,i,j);
// Fyrir:  f[i..j-1] er svæði í f
// Eftir:  Búið er að raða f[i..j-1]
void sort( double[] f, int i, int j ) {
  int p = ?1?;
  while( ?2? ) {
    // | minnstu gildi í vaxandi röð | óþekkt |
    //  i                             p        j
    int q = ?3?;
    while( ?4? ) {
      // | minnstu gildi í vax. röð | óþekkt |
      //  i                         p        j
```

```
      //     p < q <= j
      //     f[p] er minnst af f[p..q-1]
      if( f[q] ?5? f[p] )
        swap(f,p,q);
      q++;
    }
    p++;
  }
}
```

Svar:

```
// Notkun: swap(f,i,j);
// Fyrir:  f[i] og f[j] eru sæti í f
// Eftir:  Búið er að víxla gildunum í f[i] og
//         f[j]
void swap( double[] f, int i, int j );

// Notkun: sort(f,i,j);
// Fyrir:  f[i..j-1] er svæði í f
// Eftir:  Búið er að raða f[i..j-1]
void sort( double[] f, int i, int j ) {
  int p = i;
  while( p != j ) {
    // | minnstu gildi í vaxandi röð | óþekkt |
    //  i                             p        j
    int q = p+1;
    while( q != j ) {
      // | minnstu gildi í vax. röð | óþekkt |
      //  i                          p        j
      //     p < q <= j
      //     f[p] er minnst af f[p..q-1]
      if( f[q] < f[p] )
        swap(f,p,q);
      q++;
    }
    p++;
  }
}
```

3. **Spurning:** Hver er fastayrðing lykkju í ytri lykkju insertion sort?

   Svar:

```
| í vaxandi röð | óþekkt |
 i               k        j
```

4. **Spurning:** Hver eftirfarandi getur verið fastayrðing seinni lykkju heapsort?

   (a)
   ```
   | minnst í vaxandi röð | hrúga (heap) |
    0                      i              n
   ```

   (b)
   ```
   | minnst í vax. röð | uppfyllir hrúguskilyrði |
    0                  i                         n
   ```

   (c)
   ```
   | hrúga (heap) | stærst í vaxandi röð |
    0              i                     n
   ```

   Svar:

   ```
   | hrúga (heap) | stærst í vaxandi röð |
    0              i                     n
   ```

5. **Spurning:**

   (a) Hvaða fastayrðingu má nota í fyrri lykkju heapsort?

   (b) Hver er tímaflækja heapsort?

   Svar:

   (a)
   ```
   | óþekkt | uppfyllir hrúguskilyrði |
    0        k                         n
   ```
   Eða
   ```
   | hrúga | óþekkt |
    0       k        n
   ```
   Eða
   ```
   | óþekkt | uppfyllir hrúguskilyrði |
    0        k                        n
   ```
   Eða
   ```
   | hrúga | óþekkt |
    0       k      n
   ```

   (b) $O(n \log n)$ þar sem n er fjöldi sæta í fylkinu.

6. **Spurning:**

   (a) Lýsið hugmyndinni í merge-sort

   (b) Hver er tímaflækja merge-sort?

   Svar:

   (a) Til að raða runu gilda skiptum við rununni í tvær runur (óraðaðar) til helminga (u.þ.b.), röðum síðan smærri rununum og samröðum síðan þessum smærri runum til að fá raðaða heildarrunu (mergjum, *merge*). Ef upphaflega runan var styttri en tvö gildi þarf auðvitað ekkert að gera til að raða.

(b) $O(n \log n)$ þar sem n er fjöldi gilda í rununni.

7. **Spurning:** Gefið er eftirfarandi stef:

```
// Notkun:  k = skipta(f,i,j);
// Fyrir:   f[i..j-1] er ekki-tómt svæði í f
// Eftir:   Búið er að víxla gildum í svæðinu þ.a.
//          f[i..k-1] <= f[k] <= f[k+1..j-1]
int skipta( double[] f, int i, int j );
```

Skrifið quicksort stef (með lýsingu - notkun, fyrir og eftir) með hjálp þessa stefs. Ekki þarf að forrita skipta stefið.

Svar:

```
// Notkun: qsort(f,i,j);
// Fyrir:  f[i..j-1] er svæði í f
// Eftir:  Búið er að raða f[i..j-1] í vaxandi röð
void qsort( double[] f, int i, int j ) {
  if( j-i < 2 ) return;
  int k = skipta(f,i,j);
  qsort(f,i,k);
  qsort(f,k+1,j);
}
```

Spurning: Hverjar af eftirfarandi röðunaraðferðum má nota til að raða fylki með víxlunum, þ.e. án þess að geyma gildin sem verið er að raða annars staðar en í fylkinu?

8. (a) Insertion-sort
   (b) Selection-sort
   (c) Quicksort
   (d) Merge-sort
   (e) Radix-sort
   (f) Heapsort

   Svar: Allar nema merge-sort og radix-sort.

9. **Spurning:** Forritið eftirfarandi stef. Nota má einhvers konar helmingunarleit til að leysa þetta.

```
// Notkun:  x = rot(a,b,eps);
// Fyrir:   Fallið f (sem er skilgreint einhvers annars
//          staðar) er samfellt á bilinu [a,b].
//          f(a)*f(b) <= 0. eps > 0
// Eftir:   x er nálgun á rót fallsins f á bilinu
//          [a,b] með nákvæmni eps.
//          Nánar tiltekið er til z á bilinu [a,b]
```

```
//          þ.a.:
//            a) f(z) = 0
//            b) |x-z| < eps
double rot( double a, double b, double eps ) {
  ???
}
```

Svar:

```
double rot( double a, double b, double eps ) {
  if( b-a < eps ) return a;
  double m = (a+b) / 2.0;
  if( f(m)*f(a) <= 0.0 )
    return rot(a,m,eps);
  else
    return rot(m,b,eps);
}
```

10. **Spurning:** Skrifið eftirfarandi fall þannig að tímaflækjan sé viðunandi þótt veldisvísirinn sé verulega stór. Einu fleytitöluaðgerðirnar sem nota má eru þær venjulegu, þ.e. samlagning, frádráttur, margföldun og deiling.

```
// Notkun:  y = power(x,n);
// Fyrir:   n er heiltala
//          (athugið að n má vera neikvæð),
//          x er fleytitala (double)
// Eftir:   y er x^n (x í veldi n)
double power( double x, int n ) {
    ???
}
```

Svar:

```
double power( double x, int n ) {
  if( n==0 ) return 1.0;
  if( n<0 ) return 1.0 / pow(x,-n);
  if( n%2==0 )
    return pow(x*x,n/2);
  else
    return x*pow(x*x,n/2);
}
```

11. **Spurning:** Forritið eftirfarandi fall:

```
// Notkun: prim(n);
// Fyrir:  n er jákvæð heiltala
```

```
// Eftir:  Búið er að skrifa prímþætti n á
//         aðalúttak.
void prim( int n ) {
  ???
}
```

Svar:

```
void prim( int n ) {
  int p=2, m=n;
  while( m != 1 ) {
    // Allar skrifaðar tölur eru prímtölur
    // Margfeldi m og skrifaðra talna er n
    // Engin prímtala minni en p gengur upp í m
    // p >= 2
    if( m%p == 0 ) {
      cout << p << endl;
      m = m/p;
    }
    else
      p++;
  }
}
```