08.71.23/24 — Tölvunarfræði 2/2a

Sjúkra- og upptökupróf 13. júní 2008.

Engin hjálpargögn eru leyfileg.

Öll dæmi gilda jafnt.

Munið að skrifa notkunarlýsingu með forskilyrði og eftirskilyrði fyrir sérhvert stef og fastayrðingu gagna fyrir sérhverja útfærslu gagnamóts.

This is a closed-book exam.

All questions count equally.

Remember to write a usage description with precondition and postcondition for all functions and a data invariant for all implementations of data structures.

Athugið vel: Svara þarf tilskildum fjölda dæma úr hverjum hluta prófsins. Að því skilyrði uppfylltu gilda **10 bestu dæmi** til einkunnar. Þið hafið því 18 mínútur fyrir hvert dæmi að meðaltali. Byrjið því á að svara dæmum sem krefjast stuttra svara og þið getið auðveldlega svarað.

Note carefully: You need to answer the required number of questions from each part of the exam. Fulfilling that condition, the **10 best answers** count for your grade. You therefore have 18 minutes for each question on the average. You should therefore start by answering questions that require short answers and you can easily answer.

Hluti I: Röksemdafærsla

Svarið a.m.k. 1 dæmi úr þessum hluta

Part I: Programming and verification

Answer at least one question from this part.

1. Íhugið eftirfarandi lykkjumynstur:
   Consider the following loop pattern:

```
// F
for(;;) {
    // I1
    if( C1 ) break;
    S1
    // I2
    if( C2 ) continue;
    S2
    break;
```

```
}
// E
```

Hvaða samband þarf að gilda milli F, C1, I1, I2, S1, C2, S2 og E til að lykkja þessi sé rökrétt?
What relationship do we need between F, C1, I1, I2, S1, C2, S2 and E in order for this loop to be logically valid?

2. Íhugið eftirfarandi klasaskilgreiningar í Java:
Consider the following class definitions in Java:

```
class A {
    // Usage/Notkun:  y = a.sqrt(x)
    // Pre/Fyrir:     0.1 < x < 10.0
    // Post/Eftir:    x-0.01 < y*y < x+0.01
    double sqrt( double x )
    {
        ...
    }
}

class B extends A {
    // Usage/Notkun:  y = b.sqrt(x)
    // Pre/Fyrir:     0.5 < x < 1.5
    // Post/Eftir:    x-0.1 < y*y < x+0.1
    double sqrt( double x )
    {
        ...
    }
}
```

Hvernig þarf að breyta lýsingum (Notkun/Fyrir/Eftir) í klasanum B þannig að þetta verði rökrétt? Hvaða samband þarf að gilda milli lýsinganna í klasa A og klasa B?
How do we need to change the descriptions (Usage/Pre/Post) in the class B in such a way that this is logically valid? What relationship needs to hold between the descriptions in class A and class B?

3. Íhugið eftirfarandi fall, sem hefur þann tilgang að reikna x^y (`x^y`) þar sem x er fleytitala og y er heiltala:
Consider the following function, whose purpose is to compute x^y (`x^y`)

where x is a floating point number and y is an integer:

```
// Usage/Notkun: z = pow(x,y);
// Post/Eftir:   z = x^y
double pow( double x, int y )
{
    if( y<0 ) return 1.0/pow(x,-y);
    double p=1.0, q=x;
    int r=y;
    while( r != 0 ) {
        // x^y == p*q^r, r>=0
        ?1? (breytum hvorki x né y)
    }
    return(?2?);
}
```

- Hvernig getum við forritað stofninn (?1?) í lykkjuna á hraðvirkan hátt þannig að fjöldi umferða verði $O(\log y)$?
  How can we program the body of the loop (?1?) so that the loop is fast and the number of passes through the loop is $O(\log y)$?
- Hverju ættum við að skila sem gildi fallsins (?2?)?
  What value (?2?) should we return from the function?

Hluti II: Algrím

Svarið a.m.k. 3 dæmum úr þessum hluta

Part II: Algorithms

Answer at least 3 questions from this part

4. Fyllið inn þar sem spurningarmerkin eru, þ.e. skrifið hvað á að koma í stað `?1?`, o.s.frv. Þetta eru samtals fimm svör. Í stað `?2?` og `?3?` ætti að vera eitt af `<x`, `>x`, `>=x` eða `<=x`.
   Fill in for the question marks, i.e. write what should replace `?1?`, etc. This is a total of five answers. The fill-in for `?2?` and `?3?` should be one of `<x`, `>x`, `>=x` or `<=x`.

```
// Notkun: k = leita(f,i,j,x);
// Fyrir:  f[i..j-1] er í minnkandi röð og inniheldur
//         a.m.k. eitt gildi sem er minna en x og
//         a.m.k. eitt gildi sem er stærra en x.
```

```
// Eftir:  k vísar á aftasta gildi í f[i..j-1] sem
//         er stærra en x.

// Usage:  k = leita(f,i,j,x);
// Pre:    f[i..j-1] is in descending order and
//         contains at least one value less
//         than x and at least one value greater
//         than x.
// Post:   k is an index to the last (highest indexed)
//         value in f[i..j-1] which is greater than x.

int leita( double[] f, int i, int j, double x ) {
  int p=i, q=j;
  while( ?1? ) {
    // | ?2? | óþekkt/unknown | ?3? |
    //  i     p                q     j
    int m = (p+q)/2;
    if( f[m] > x )
      p = ?4?;
    else
      q = ?5?;
  }
  return p-1;
}
```

5. Gefið er eftirfarandi stef eða aðferð:
   The following function or method is given:

```
// Notkun:  k = split(f,i,j);
// Fyrir:   f[i..j-1]  er svæði í f.
// Eftir:   i <= k < j og búið er að umraða gildum í
//          svæðinu f[i..j-1] þannig að
//           f[i..k-1] <= f[k] <= f[k+1..j-1].

// Usage:   k = split(f,i,j);
// Fyrir:   f[i..j-1] is an area in f.
// Eftir:   i <= k < j and the values in the area
//          f[i..j-1] have been permuted in such a
//          way that
//           f[i..k-1] <= f[k] <= f[k+1..j-1].
```

```
int split( double[] f, int i, int j );
```

Skrifið quicksort stef (með lýsingu - notkun, fyrir og eftir) með hjálp þessa stefs. Ekki þarf að forrita split stefið.
Write a quicksort function (with description - usage, pre, post) using this function. You do not need to program the split function.

6. Tilgreinið tímaflækju eftirfarandi röðunaraðferða miðað við að raðað sé n slembitölum. Tilgreinið *bæði* versta tíma og meðaltíma.
Specify the time complexity of the following sorting methods assuming that we are sorting n random numbers. Specify *both* worst-time and average-time complexity.

   (a) Heapsort

   (b) Merge-sort

   (c) Quicksort

   (d) Insertion-sort

   (e) Radix-sort

   (f) Röðum með því að moka gildunum í AVL tré (sem upphaflega er tómt) og moka þeim síðan út í vaxandi röð
   Sort by shovelling the numbers into an AVL tree (which is originally empty) and then shovelling them out in ascending order

   (g) Röðum með því að moka gildunum í Splay tré (sem upphaflega er tómt) og moka þeim síðan út í vaxandi röð
   Sort by shovelling the numbers into a Splay tree (which is originally empty) and then shovelling them out in ascending order

7. Lýsið heapsort. Tilgreinið tímaflækju heapsort og rökstyðjið hana.
Describe heapsort. Specify the time complexity of heapsort and justify deducing that this is the complexity.

8. Íhugið eftirfarandi stef.
Consider the following method.

```
// Notkun: partition(f,i,j,k);
// Fyrir:  i <= k < j, f[i..j-1] er svæði í f
// Eftir:  Búið er að umraða svæðinu f[i..j-1] þ.a.
//         f[i..k-1] <= f[k] <= f[k+1..j]

// Usage:  partition(f,i,j,k);
```

```
// Pre:    i <= k < j, f[i..j-1] is an area in f
// Eftir:  The values in the area f[i..j-1] have been
//         permuted in such a way that
//         f[i..k-1] <= f[k] <= f[k+1..j]

void partition( double f[], int i, int j, int k ) {
    ...
}
```

- Sýnið hvernig nota má þetta stef til að finna miðgildi 99 gilda sem geymd eru í fylki g í sætum g[0..98].
  Show how to use this method to find the median value of 99 values stored in an array g in positions g[0..98].
- Sýnið hvernig forrita má þetta stef. Full stig fást aðeins ef útfærslan hefur meðaltímaflækju $O(n)$ fyrir n mismunandi slembigildi.
  Show how to program this method. Full points will only be given if the implementation has average time complexity $O(n)$ for n different random values.

Hluti III: Gagnamót

Svarið a.m.k. 3 dæmum úr þessum hluta

Part III: Data Structures

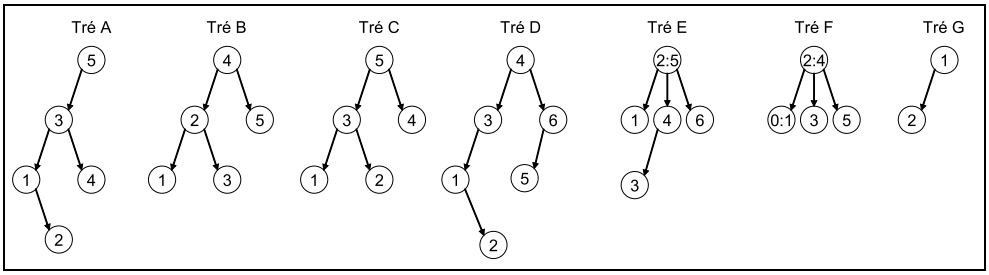
Answer at least 3 questions from this part.

9. Íhugið eftirfarandi myndir af trjám. Segið til fyrir hverja mynd hvort hún getur staðið fyrir eitt eða fleiri af eftirfarandi:

   - Tvíleitartré – Binary search tree
   - AVL-tré – AVL tree
   - Splay-tré – Splay Tree
   - Rautt-svart tré (ef svo er, tilgreinið þá einnig hvaða hnúta má mála rauða til að það gangi upp)
     Red-black tree (if so, specify which nodes you could paint red in order for this to be true).
   - 2-3 tré – 2-3 tree
   - Hrúga með hæsta gildi efst – Heap with greatest value on top
   - Hrúga með minnsta gildi efst – Heap with least value on top

Athugið að hér er ætlast til að öll tvíleitartré séu með gildin í vaxandi *in-order* röð.
Note that we want all binary search trees to be in ascending in-order order.

Athugið einnig að sama mynd af tré getur vel staðið fyrir fleiri en eina af þessum upp töldu gerðum af trjám.
Note also that the same picture of a tree can easily stand for more than one of these listed types of trees.



10. Hver er fastayrðing gagna fyrir AVL tré?
    What is the data invariant (representation invariant) for AVL trees?

11. Í tætitöflum og í fleiri vandamálum eru notuð stækkanleg fylki sem hafa sömu aðgerðir og venjuleg fylki en auk þess aðgerð til að bæta einu sæti aftast á fylkið. Hver er tímaflækja þessarar aðgerðar og hvers konar tímaflækja er það (versta tilfelli, meðaltal eða innistæðubungin)? Lýsið því hvernig aðgerðin er útfærð til að ná þessari tímaflækju.
    In hash tables and in other tasks we use extendible arrays that have the same operations as ordinary arrays but also have an operation for appending one position to the end of the array. What is the time complexity of this operation and what kind of time complexity are we talking about (worst case, average or amortized)? Describe how to implement this operation in order to achieve this time complexity.

12. Lýsið tímaflækju splay trjáa. Gerið skilmerkilega grein fyrir merkingu þess að tímaflækjan er innistæðubundin frekar en t.d. meðaltímaflækja eða versta tímaflækja. Að hvaða leyti eru splay tré hraðvirkari en AVL tré? Að hvaða leyti eru þau hægvirkari?
    Describe the time complexity of splay trees. Describe precisely what it means that the complexity is amortized rather than, for example, average complexity or worst case complexity. In what sense are splay trees faster than AVL trees? In what sense are they slower?

13. Skrifið klasa í Java eða C++ fyrir biðröð heiltalna. Þið megið sleppa því að forrita boðin, nema fyrir aðferðina til að bæta gildi í biðröðina, en munið að hafa notkun, forskilyrði og eftirskilyrði fyrir öll boð og munið að hafa skýra fastayrðingu gagna.
    Write a class in Java or C++ for queues of integers. You do not need to program the methods, except for the method for adding a value to the queue, but remember to have usage, pre and port for all messages (methods), and remember to have a clear data invariant.

Hluti IV: Blandað efni

Ekki þarf endilega að svara neinu dæmi úr þessum hluta, en ekki gleyma að svara 10 dæmum í heild.

Part IV: Miscellaneous

You do not need to answer any question from this part, but do not forget to answer a total of 10 questions.

14. Skrifið stef sem skilar *stærsta* prímtöluþætti tölu sem skal vera viðfang stefsins.
    Write a function that returns the *largest* prime factor of a number that should be an argument to the function.

15. Skrifið forrit sem les strengi frá aðalinntaki og skrifar strengina í öfugri lesröð á aðalúttak. Til dæmis, ef forritið les línurnar "aa", "bb", "aa", "cc" þá skal það skrifa línurnar "cc", "aa", "bb", "aa".
    Write a program that reads strings from standard input and writes the strings in reverse order to standard output. For example, if the program reads the lines "aa", "bb", "aa", "cc" then it should write the lines "cc", "aa", "bb", "aa".

16. Skrifið forrit sem les ótakmarkaðan fjölda fleytitalna (double) og skrifar meðaltal þeirra.
    Write a program that reads an unlimited number of floating point numbers (double) and writes their average.