TÖL212M Rökstudd forritun

Prófdagur og tími: 24.04.2019 09:00-12:00

Prófstaður:
VR-2 - V258 (fjöldi: 13)

HÁSKÓLI ÍSLANDS

Iðnaðarverkfræði-, vélaverkfræði- og tölvunarfræðideild

Kennari:
Snorri Agnarsson (snorri@hi.is / S: 8613270 / GSM: 8613270) Umsjónarkennari

Skriflegt próf

Skráðir til prófs: 13

Kennslumisseri: Vor 2019

Úrlausnir skulu merktar með nafni

Prófbók/svarblöð:
Prófbók óþörf

Hjálpargögn:
Engin

Önnur fyrirmæli:

Aðgangur að prófverkefni að loknu prófi:
Kennslusvið sendir eintak í prófasafn

Einkunnir skulu skráðar í Uglu eigi síðar en 08.05.2019.

ATHUGIÐ að einhverjar úrlausnir úr fjölmennum prófum geta verið í þunnum umslögum sem auðvelt er að yfirsjást. GÓÐ VINNUREGLA er að byrja á því að opna öll umslög, telja úrlausnir og athuga hvort fjöldi stemmir við uppgefinn fjölda sem kvittað var fyrir.

TÖL212M Lokapróf

Nafn/Name: .....

Háskólatölvupóstfang/University Email: .....

1. Engin hjálpargögn eru leyfileg.
2. Skrifið svörin á þessar síður, ekki á önnur blöð og ekki á baksíður.
3. Ef svarið kemst ekki fyrir á tilteknu svæði má skrifa á auðar síður aftast, en þá skalt þú láta vita af því með því að skrifa tilvísun í tiltekið svæði, til dæmis "framhald á blaðsíðu 14".
4. Forðist að skemma eða rífa þessar síður, þær þurfa að fara gegnum skanna. Skrifið skýrt með **dökku letri** og ekki skrifa í spássíur.
5. Baksíður **verða ekki skannaðar** og má nota fyrir krass. **Ekki verður tekið tillit til** svara sem skrifuð eru á baksíður.
6. Prófið skiptist í **hluta**. Svarið **8** spurningum í heild og að minnsta kosti **tilteknum lágmarksfjölda** í hverjum hluta.
7. Ef þú svarar fleiri en **8** spurningum þá verður einkunn þín reiknuð sem **meðaltal allra svara** nema þú **krossir skýrt út** svör sem þú vilt ekki að gildi. Þú verður að krossa út allt svarið, ekki aðeins hluta þess.
8. Munið að öll Dafny föll þurfa **notkunarlýsingu** með **requires/ensures**. Allar lykkjur þurfa **invariant** sem dugar til að rökstyðja.
9. Munið að öll Java föll þurfa **notkunarlýsingu** með Notkun/Fyrir/Eftir. Allar lykkjur þurfa **fastayrðingu** sem dugar til að rökstyðja.
10. Munið að nota viðeigandi **innfellingu** í öllum forritstexta.

Hluti I – Helmingunarleit o.fl.

Svarið að minnsta kosti tveimur spurningum í þessum hluta – Munið að svara a.m.k. 8 spurningum í heild

1.

Skrifið fall í Dafny sem leitar með helmingunarleit í heiltalnafylki, a, sem raðað er í vaxandi röð, að aftasta sæti sem inniheldur gildi < 100. Ef ekkert slíkt sæti er til skal skila -1.

Svar:

2.

Skrifið endurkvæmt helmingunarleitarfall í Dafny sem leitar í svæði í heiltalnafylki sem raðað er í minnkandi röð að fremsta sæti innan svæðisins sem inniheldur gildi < 100. Ef ekkert slíkt sæti er til innan svæðisins skal skila -1.

Svar:

3.

Skrifið endurkvæmt helmingunarleitarfall í Java sem hefur eftirfarandi lýsingu:

```
// Notkun: int k = find(a,i,n,x);
// Fyrir:  0 <= i <= i+n <= a.length, x er heiltala,
//         a[i..i+n) er í vaxandi röð. (Í Dafny myndum við
//         skrifa a[i..i+n] í stað a[i..i+n).)
//         n >= 0 og n+1 er veldi af tveimur, þ.e. lögleg
//         gildi fyrir n eru 0,1,3,7,15,31, o.s.frv.
// Eftir:  i <= k <= i+n.
//         a[i..k) < x <= a[k..i+n).
//         Fylkið a er óbreytt.
static int find( int[] a, int i, int n, int x )
{
  ...
}
```

Svar:

Hluti II – Quicksort o.fl.

Svarið að minnsta kosti einni spurningu í þessum hluta – Munið að svara a.m.k. 8 spurningum í heild

4.

Gerið ráð fyrir að til sé Dafny fall með eftirfarandi lýsingu:

```
method Partition( a: array<int>, i: int, j: int )
      returns ( p: int, q: int )
   modifies a;
   requires 0 <= i < j <= a.Length;
   ensures i <= p < q <= j;
   ensures forall r | p <= r < q :: a[r] == a[p];
   ensures forall r | i <= r < p :: a[r] < a[p];
   ensures forall r | q < r < j :: a[r] > a[p];
   ensures multiset(a[i..j]) == old(multiset(a[i..j]));
   ensures a[..i] == old(a[..i]);
   ensures a[j..] == old(a[j..]);
```

Skrifið Quicksort fall sem notar þetta fall sem hjálparfall. Ekki forrita Partition fallið hér.

Svar:

5.

Forritið fallið Partition sem lýst er að ofan. Munið að allar lykkjur þurfa invariant.

Svar:

6.

Gerið ráð fyrir að til sé Dafny fall með eftirfarandi lýsingu:

```
method SemiPartition( a: array<int>, i: int, k: int, j: int )
   modifies a;
   requires 0 <= i <= k <= j <= a.Length;
   ensures forall p,q | i <= p < k <= q < j :: a[p] <= a[q];
   ensures multiset(a[i..j]) == old(multiset(a[i..j]);
   ensures a[..i] == old(a[..i]);
   ensures a[j..] == old(a[j..]);
```

Skrifið Quicksort fall sem notar þetta fall sem hjálparfall. Ekki forrita SemiPartition fallið. Takið eftir að k er hér ekki skilagildi heldur viðfang (argument). Takið einnig eftir að fallið skiptir svæðinu í tvennt, ekki þrennt.

Svar:

Hluti III – Tvíleitartré

Svarið að minnsta kosti tveimur spurningum í þessum hluta – Munið að svara a.m.k. 8 spurningum í heild

7.

Gerið ráð fyrir skilgreiningunni

```
datatype BST = BSTEmpty | BSTNode(BST,int,BST)
```

eins og í skránni okkar BST.dfy. Gerið einnig ráð fyrir föllunum IsTreePath, TreeSeq, PreSeq, MidSeq PostSeq, PreSeqIncluding, o.s.frv. (spyrjið ef þið munið ekki nöfnin). Skrifið fall sem leitar með lykkju í tvíleitartré og skilar tilvísun á aftasta hnút í milliröð sem inniheldur gildi <100, eða skilar BSTEmpty ef slíkur hnútur finnst ekki í leitartrénu. Munið að skrifa fulla lýsingu með requires/ensures og skrifa invariant fyrir lykkjuna.

Svar:

8. **Leysið sama vandamál og að ofan, en núna í Java og notið nú endurkvæmni en ekki lykkju. Notið gamalkunna skilgreiningu á trjáhnútum, þ.e.:**

```
public class BSTNode {
  private BSTNode left,right;
  private int val;
  public BSTNode( BST a, int x, BST b )
  { left=a; val=x; right=b; }
  public static left( BSTNode t ) { return t.left; }
  public static right( BSTNode t ) { return t.right; }
  public static rootValue( BSTNode t ) { return t.val; }
}
```

(Ég sleppi hér Notkun/Fyrir/Eftir til að spara pláss því þau eru gamalkunnug og augljós. Athugið að það þýðir ekki að nemendur megi sleppa að skrifa gamalkunnugar og augljósar lýsingar fyrir sína klasa.)

Svar:

9. **Skrifið fall í Dafny sem skilar minnsta gildi í tvíleitartré sem ekki má vera tómt. Þið megið nota lykkju eða endurkvæmni, að því tilskildu að rökstuðningur sé réttur (þ.e. rétt requires, ensures og invariant).**

   Svar:

10.

Skrifið fall (method eða function method) SumTree í Dafny sem skilar summu allra talnanna í tvíundartré. Summa talna í runu s er skilgreind sem SumSeq(s) þar sem fallið SumSeq er skilgreint með

```
function SumSeq( s: seq<int> ): int
{
  if s == [] then
    0
  else
    s[0]+SumSeq(s[1..])
}
```

Fallið SumTree þarf að hafa rétta lýsingu sem tryggir að skilagildið sé rétt. Óþarfi er að Dafny kerfið geti sannað fallið, en það þarf samt að vera rétt. Í lýsingunni megið þið kalla á fallið SumSeq í viðeigandi ensures klausu.

Svar:

Hluti IV – Ýmislegt

Svarið að minnsta kosti einni spurningu í þessum hluta – Munið að svara a.m.k. 8 spurningum í heild

10.

Forritið stofninn á fallinu Mul að neðan. Ekki nota lykkju, ekki nota margföldunarvirkjann * og sjáið til þess að fallið virki fyrir risastórar tölur á sæmilegum hraða. Notið endurkvæmni og enga lykkju. Dýpt endurkvæmninnar ætti að vera $O(\log(y))$. Þið megið leggja saman heiltölur (þ.e. $a + b$) og helminga heiltölur (þ.e. $a/2$) og athuga hvort heiltala er oddatala eða slétt (þ.e. $a\%2$).

```
method Mul ( x: int, y: int ) returns ( p: int )
  decreases y;
  requires x >= 0 && y >= 0;
  ensures p == x*y;
```

Svar:

11.

Skrifið Java fall sem tekur int[] sem viðfang og skilar stærstu tölunni í fylkinu. Munið að setja viðeigandi lýsingar og fastayrðingar og sjáið til þess að forskilyrðið tryggi að til sé stærsta tala.

Svar:

(auð blaðsíða/empty page)

(auð blaðsíða/empty page)