TÖL304G Forritunarmál

Prófdagur og tími: 18.12.2012 13:30-16:30

Prófstaður:
Aðalbygging - A-231, MoN (fjöldi:5)
Háskólatorg - HT-302 - tölvuver, Aðgengissetur (fjöldi:2)
VR-2 - V02-138 (fjöldi:14)
VR-2 - V02-152 (fjöldi:24)
VR-2 - V02-155 (fjöldi:24)
VR-2 - V02-156 (fjöldi:22)
VR-2 - V02-258 (fjöldi:11)
VR-2 - V02-261 (fjöldi:15)

HÁSKÓLI ÍSLANDS

Iðnaðarverkfræði-, vélaverkfræði- og tölvunarfræðideild

Skriflegt próf

Skráðir til prófs: 117

Kennarar:
Kristján Jónasson (jonasson@hi.is / S: 4735 / GSM: 8228860) Umsjónarmaður
Snorri Agnarsson (snorri@hi.is / S: 8613270 / GSM: 8613270) Umsjónarkennari
Ívar Haukur Sævarsson (Ekkert netfang / GSM: 7774487) Aðstoðarkennari
Guðmundur Björn Birkisson (Ekkert netfang / GSM: 8467460) Aðstoðarkennari

Kennslumisseri: Haust 2012

Úrlausnir skulu merktar með nafni

Prófbók/svarblöð:
Línustrikuð prófbók

Hjálpargögn:
Engin hjálpargögn eru leyfileg.

Önnur fyrirmæli:

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

Einkunnir skulu skráðar í Uglu eigi síðar en 08.01.2013.

AHUGIÐ 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.

TÖL304G Programming Languages
Final Exam Autumn 2012

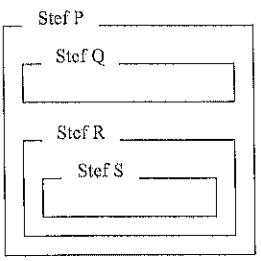
Answer 10 of the following questions and at least two from each of chapters I–III. If you answer more than 10 questions you should specify which answers should count toward your grade. If you do not specify such answers and answer more than 10 then the average will be taken from all your answer attempts. All questions count equally. **Write on the foremost white page in your solution book which solutions should count toward your grade.**

No help materials are allowed.

Part I: Block-Structure etc.

1. Describe how activation records in block-stuctured programming languages are linked together with control links (dynamic links) and access links (static links), and how links are used to access variables in different nesting levels.

2. Assume nesting in a block-structured programming language as shown in the figure. Which of the following statements are then true and which are false? One wrong answer causes the grade for the question to be zero.



(a) S may be called from P.

(b) S may be called from Q.

(c) S may be called from R.

(d) P may be called from R.

(e) P may be called from S.

(f) P may be called from Q.

(g) Q may be called from R.

(h) Q may be called from S.

(i) Q may be called from P.

(j) R may be called from Q.

(k) R may be called from S.

(l) R may be called from P.

(m) In P you can use local variables in Q.

(n) In P you can use local variables in R.

(o) In P you can use local variables in S.

(p) In Q you can use local variables in P.

(q) In Q you can use local variables in R.

(r) In Q you can use local variables in S.

(s) In S you can use local variables in P.

(t) In S you can use local variables in R.

(u) In S you can use local variables in Q.

(v) In R you can use local variables in P.

(w) In R you can use local variables in S.

(x) In R you can use local variables in Q.

3. Give arguments that support the statement that in order for a block-structured programming language allow a function to be returned as a value from a function then activation records should be stored on the heap rather than on a stack. Describe the problems that arise if activation records are on a stack.

4. Answer all of the following:

   (a) What information is stored in activation records?

   (b) Which of these information items are only in block-structured programming languages and why?

   (c) Name two programming languages where activation records are usually stored on the stack and two where the activation records are usually on the heap.

(d) Describe why, in the latter two programming languages, it is necessary for the activation records to be stored on the heap.

5. The following program text is in some imagined programming language. (Ask if you feel the syntax or semantics is not obvious.)

```
void f(x,y)
{
  x = 2;
  print x,y;
  y = 1;
}

int i,a[10];
for( i=0 ; i!=10 ; i++ ) a[i]=i;
f(a[0],a[a[0]]);
print a[0], a[1], a[2];
```

What does this program write (five values each time) is the arguments are

- passed by value?
- passed by reference?
- passed by name?

Part II: List Processing etc.

6. Write a function in Scheme, CAML, Morpho or Haskell that takes one argument that is a list of lists of floating point numbers between 0 and 1 and returns a number that is the lowest maximum of the inner lists, i.e. the lowest among the numbers that result from taking the maximum of each inner list. You may assume that the maximum of the empty set is 0 and the minimum of the empty set is 1.

7. Write a tail-recursive function in Scheme, Haskell, CAML or Morpho that takes one integer argument, $n \geq 0$, and returns the list $(1\ 2\ \ldots n)$. If the function is not tail-recursive only half the points are given for the answer.

8. Write a tail recursive[1] function in Scheme, CAML or Morpho, that takes as argument one integer $n \geq 0$ and returns the list $[n, n-1, \ldots, 1]$.

[1] It is enough that the implementation use a tail recursive help function to restrict the depth of the recursion. You may not do a loop implementation in Morpho.

9. Program a stream in Scheme or Morpho that contains the infinite series $[1, 3, 5, 7, \ldots]$ of all positive odd numbers. If you need helper functions you need to program these as well.

10. Write two functions in Scheme, CAML, Morpho or Haskell that are traditional implementations of insert-left and insert-right, i.e. functions that take three arguments and apply a binary operation from the right or from the left.

Part III: Modular Programming etc.

11. Assume the existence of classes A and B in some object-oriented programming language and that B is a subclass of A and that both classes have a message t with the following descriptions in A and B.

    - Description in A:

      Usage: y = z.t(x);
      Pre: $a \leq x \leq b$.
      Post: $c \leq y \leq d$.
    - Description in B:

      Usage: y = z.t(x);
      Pre: $e < x < f$.
      Post: $g < y < h$.

    What does the relationship between a, b, c, d, e, f, g, and h in order for the logical relationship between the classes A and B to be correct? The relationship you specify must be necessary and sufficient.

12. Write a design document for a module in Morpho, Java or C++ that implements a queue of values (or objects). The module should be polymorphic (fjölnota) enough so that you can make a queue of any values that fulfill reasonable constraints, but the module should also use the typing constraints of the programming language to ensure as far as reasonable that the queue only contains values of the relevant type.

    Remember information hiding. Only a design document is needed, no implementation is needed.

13. Write a design document for a polymorphic module for priority queues in Java, C++ or Morpho.

    Remember information hiding.

    Also write a data invariant (fastayrðing gagna) and program the method to remove a value from the priority queue.

14. Write a design document for a polymorphic module in Morpho that implements arithmetic on complex numbers. Exports from the module should include operations for computing with complex numbers and imports should include operations for arithmetic on real numbers.

Part IV: Miscellaneous

15. Describe the reference count garbage collection method. Name some reason why reference count is usually slower than other methods. Name a reason why in spite of the slow speed reference count is often used in programming languages such as C++.

16. Consider the following BNF grammar:

```
<E> ::= <T> <Em>
<Em> ::= + <T> <Em>
<Em> ::= ϵ
<T> ::= <F> <T>
<T> ::= ϵ
<F> ::= <F> !
<F> ::= ( <E> )
<F> ::= t
<F> ::= f
```

Which of the following strings are in the language (one wrong answer gives grade zero)?

(a) `f+t`

(b) `!f+t`

(c) `f!+t`

(d) `f+!t`

(e) `f+t!`

(f) `ft`

(g) `!ft*t`

(h) `ft!*t`

(i) `f!t`

(j) `ft!`

(k) `(ft)`

(l) `(f+t)!`

(m) `(f+tf)!`

17. Draw a finite state machine (deterministic or nondeterministic) for the language over the alphabet $\{(,)\}$ that contains the strings that have balanced parentheses and the nesting depth is at most 3. The strings '(())' and '()()(()())' should be in the language, but not '((()))'.

18. What is the set intersection of the languages with the following BNF grammars?

Language A:

```
<C> ::= <B> <A>
<B> ::= c <B> b
<B> ::= ϵ
<A> ::= a <C>
<A> ::= ϵ
```

Language B:

```
<C> ::= <B> <A>
<B> ::= c <B>
<B> ::= ϵ
<A> ::= b <C> a
<A> ::= ϵ
```

19. Show BNF, EBNF or syntax diagrams for a language og λ-expressions with parentheses, the variable names a, b and c, function applications NM where N and M are λ-expressions, and function definitions $\lambda x.N$ where x is one of the variable names and N is a λ-expression. Examples of legal expressions are $\lambda a.ab$, $a(ab)$, abc and $(\lambda a.ba)c$, but not, for example x or $\lambda ab.c$.

20. Consider the languages defines on one hand by the regular expression $(x(x(x(xy)^*y)^*y)^*y)^*$ and on the other hand by the BNF grammar below. Note that ϵ stands for the empty string.

S ::= x <S> y <S> | ϵ

(a) Specify whether the regular expression and the BNF grammar describe the same language.

(b) If the languages are not the same then describe both languages in a short text and show a string that is in one of the languages but not the other.

(c) If the languages are the same then describe the language that they describe in a short text.

TÖL304G Forritunarmál
Lokapróf haustið 2012

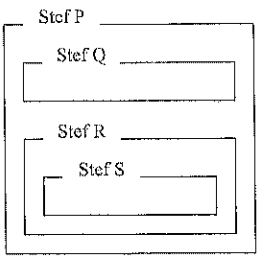
Svarið 10 af eftirfarandi dæmum og a.m.k. tveimur úr hverjum af köflum I–III. Ef þið svarið fleiri en 10 dæmum skuluð þið tilgreina hvaða svör eiga að gilda til einkunnar. Ef þið tilgreinið ekki slík svör og svarið fleiri en 10 verður tekið meðaltal allra þeirra dæma sem þið reynið við. Öll dæmi gilda jafnt. **Skrifið á fremstu hvítu blaðsíðu úrlausnarbókar hvaða lausnir eiga að gilda til einkunnar.**

Engin hjálpargögn eru leyfileg.

Hluti I: Bálkmótun o.fl.

1. Lýsið því hvernig vakningarfærslur í bálkmótuðum forritunarmálum eru tengdar saman með stýrihlekkjum (*control link, dynamic link*) og tengihlekkjum (*aðgangshlekkjum, access link, static link*) og hvernig hlekkir eru notaðir til að gefa aðgang að breytum í mismunandi földunarhæðum.

2. Gerið ráð fyrir földun í bálkmótuðu forritunarmáli eins og myndin sýnir. Hverjar eftirfarandi fullyrðinga eru þá sannar og hverjar eru ósannar? Eitt rangt svar veldur því að ekkert fæst fyrir dæmið.



(a) Kalla má á S úr P.

(b) Kalla má á S úr Q.

(c) Kalla má á S úr R.

(d) Kalla má á P úr R.

(e) Kalla má á P úr S.

(f) Kalla má á P úr Q.

(g) Kalla má á Q úr R.

(h) Kalla má á Q úr S.

(i) Kalla má á Q úr P.

(j) Kalla má á R úr Q.

(k) Kalla má á R úr S.

(l) Kalla má á R úr P.

(m) Í P má nota staðværar breytur úr Q.

(n) Í P má nota staðværar breytur úr R.

(o) Í P má nota staðværar breytur úr S.

(p) Í Q má nota staðværar breytur úr P.

(q) Í Q má nota staðværar breytur úr R.

(r) Í Q má nota staðværar breytur úr S.

(s) Í S má nota staðværar breytur úr P.

(t) Í S má nota staðværar breytur úr R.

(u) Í S má nota staðværar breytur úr Q.

(v) Í R má nota staðværar breytur úr P.

(w) Í R má nota staðværar breytur úr S.

(x) Í R má nota staðværar breytur úr Q.

3. Rökstyðjið að til þess að bálkmótað forritunarmál bjóði þann möguleika að falli sé skilað úr falli þurfi vakningarfærslur að vera geymdar í kös (heap) frekar en á hlaða (stack). Lýsið þeim vandamálum sem upp koma ef vakningarfærslur eru á hlaða.

4. Svarið öllum eftirfarandi:

   (a) Hvaða upplýsingar eru geymdar í vakningarfærslum forritunarmála?

   (b) Hverjar þessara upplýsinga eru aðeins í bálkmótuðum forritunarmálum og hvers vegna?

   (c) Nefnið tvö forritunarmál þar sem vakningarfærslur eru yfirleitt geymdar á hlaðanum og tvö þar sem vakningarfærslur verða að vera geymdar í kös.

   (d) Rökstyðjið að í síðarnefndu tveimur forritunarmálunum sé nauðsynlegt að vakningarfærslur séu í kös.

5. Eftirfarandi forritstexti er í einhverju ímynduðu forritunarmáli. (Spyrjið ef ykkur finnst merking eða málfræði ekki augljós.)

```
void f(x,y)
{
  x = 2;
  print x,y;
  y = 1;
}

int i,a[10];
for( i=0 ; i!=10 ; i++ ) a[i]=i;
f(a[0],a[a[0]]);
print a[0], a[1], a[2];
```

Hvað skrifar þetta forrit (fimm gildi í hvert skipti) ef viðföngin eru

- gildisviðföng?
- tilvísunarviðföng?
- nafnviðföng?

Hluti II: Listavinnsla o.fl.

6. Skrifið fall í Scheme, CAML, Morpho eða Haskell sem tekur eitt viðfang sem er listi lista af fleytitölum milli 0 og 1 og skilar tölu sem er minnsta hágildi innri listanna, þ.e. minnst af þeim tölum sem fást þegar fundin er hæsta tala í hverjum innri lista. Þið megið reikna með því að hæsta gildi í tómamenginu sé 0 og lægsta gildi í tómamenginu sé 1.

7. Skrifið halaendurkvæmt fall í Scheme, Haskell, CAML eða Morpho, sem tekur eina heiltölu $n \geq 0$ sem viðfang og skilar listanum $(1\ 2\ \ldots n)$. Ef fallið er ekki halaendurkvæmt fæst aðeins hálft fyrir dæmið.

8. Skrifið halaendurkvæmt[1] fall í Scheme, CAML eða Morpho, sem tekur eina heiltölu $n \geq 0$ sem viðfang og skilar listanum $[n, n-1, \ldots, 1]$.

9. Forritið straum í Scheme eða Morpho sem inniheldur óendanlegu rununa $[1, 3, 5, 7, \ldots]$ af öllum jákvæðum oddatölum. Ef ykkur vantar hjálparföll verðið þið að skilgreina þau.

[1] Það dugar að útfærslan noti halaendurkvæmt hjálparfall til að takmarka dýpt endurkvæmninnar. Það má ekki leysa þetta með lykkju í Morpho.

10. Skrifið tvö föll í Scheme, CAML, Morpho eða Haskell sem eru hefðbundnar útfærslur á insert-left og insert-right, þ.e. föll sem taka þrjú viðföng og beita tvíundaraðgerð frá vinstri til hægri eða frá hægri til vinstri.

Hluti III: Einingaforritun o.fl.

11. Gerið ráð fyrir að til séu klasar A og B í einhverju hlutbundnu forritunarmáli og að B sé undirklasi A og að báðir klasarnir innihaldi tilviksboð t með eftirfarandi lýsingar í A og B.

    - Lýsing í A:

      Notkun: y = z.t(x);
      Fyrir: $a \leq x \leq b$.
      Eftir: $c \leq y \leq d$.
    - Lýsing í B:

      Notkun: y = z.t(x);
      Fyrir: $e < x < f$.
      Eftir: $g < y < h$.

    Hvað þarf sambandið að vera milli a, b, c, d, e, f, g og h til að rökfræðilegt samband klasanna A og B sé rétt? Sambandið sem þið lýsið skal vera nauðsynlegt og nægjanlegt.

12. Skrifið hönnunarskjal fyrir einingu í Morpho, Java eða C++ sem útfærir biðröð (*queue*) gilda (eða hluta). Einingin skal vera nægilega fjölnota til að unnt sé að búa til biðröð hluta af hvaða tagi sem er sem uppfyllir eðlileg skilyrði, en þó skal nýta þá tögun sem forritunarmálið býður upp á til að unnt sé að tryggja að biðröðin innihaldi aðeins gildi af því tagi sem til er ætlast.

    Munið upplýsingahuld. Aðeins þarf hönnunarskjal, enga útfærslu þarf.

13. Skrifið hönnunarskjal fyrir fjölnota einingu fyrir forgangsbiðraðir í Java, C++ eða Morpho.

    Munið upplýsingahuld.

    Skrifið einnig fastayrðingu gagna og útfærið aðgerðina til að fjarlægja gildi úr forgangsbiðröðinni.

14. Skrifið hönnunarskjal fyrir fjölnota einingu í Morpho sem útfærir tvinntölureikninga. Úr einingunni skulu vera útfluttar aðgerðir til að reikna með tvinntölur og innfluttar skulu vera aðgerðir til reikninga með rauntölur.

Hluti IV: Blandað efni

15. Lýsið ruslasöfununaraðferðinni sem byggist á tilvísunartalningu. Nefnið einhverja ástæðu þess að tilvísunartalning er oftast hægvirkari en aðrar aðferðir. Nefnið ástæðu þess að þrátt fyrir lélegan hraða er tilvísunartalning oft notuð í forritunarmálum s.s. C++.

16. Gefin er eftirfarandi BNF mállýsing:

```
<E> ::= <T> <Em>
<Em> ::= + <T> <Em>
<Em> ::= ϵ
<T> ::= <F> <T>
<T> ::= ϵ
<F> ::= <F> !
<F> ::= ( <E> )
<F> ::= t
<F> ::= f
```

Hverjir eftirfarandi strengja eru í málinu (eitt rangt svar gefur núll)?

(a) `f+t`
(b) `!f+t`
(c) `f!+t`
(d) `f+!t`
(e) `f+t!`
(f) `ft`
(g) `!ft*t`
(h) `ft!*t`
(i) `f!t`
(j) `ft!`
(k) `(ft)`
(l) `(f+t)!`
(m) `(f+tf)!`

17. Teiknið endanlega stöðuvél (löggenga eða brigðgenga) fyrir málið yfir stafrófið $\{(,)\}$ sem inniheldur þá strengi sem hafa sviga í jafnvægi og dýpt sviga er í mesta lagi 3. Strengirnir '(())' og '()()((()))' eiga að vera í málinu en ekki '((()))'.

18. Hvað er sniðmengi málanna sem hafa eftirfarandi BNF mállýsingar?

Mál A:

```
<C> ::= <B> <A>
<B> ::= c <B> b
<B> ::= ϵ
<A> ::= a <C>
<A> ::= ϵ
```

Mál B:

```
<C> ::= <B> <A>
<B> ::= c <B>
<B> ::= ϵ
<A> ::= b <C> a
<A> ::= ϵ
```

19. Sýnið BNF, EBNF eða málrit fyrir mál λ-reiknisegða með svigum, breytunöfnunum a, b og c, fallsbeitingu NM þar sem N og M eru λ-reiknisegðir, og fallsskilgreiningum $\lambda x.N$ þar sem x er eitt breytunafnanna og N er λ-reiknisegð. Dæmi um löglegar formúlur eru þá t.d. $\lambda a.ab$, $a(ab)$, abc og $(\lambda a.ba)c$, en ekki t.d. x eða $\lambda ab.c$.

20. Íhugið málin sem skilgreind eru annars vegar með reglulegu segðinni $(x(x(x(xy)^*y)^*y)^*y)^*$ og hins vegar með BNF mállýsingunni (athugið að ϵ stendur fyrir tóma strenginn) að neðan.

    S ::= x <S> y <S> | ϵ

    (a) Segið til um hvort reglulega segðin og BNF mállýsingin lýsa sama máli.

    (b) Ef málin eru ekki þau sömu, lýsið þá báðum málunum í stuttum texta og sýnið streng sem er í öðru málinu en ekki hinu.

    (c) Ef málin eru þau sömu lýsið þá málinu sem þau lýsa í stuttum texta.