TIE-02500 Rinnakkaisuus

Tentti 26.8.2015

Tentin vastuuhenkilö: jyke.savia@tut.fi
Laskimen käyttö on kiellettyä.
Tenttiohjesäännön 2§ mukainen ilmoitus: tenttipaperia ei tarvitse palauttaa, mutta siitä ei saa tehdä paperilennokkia eikä sillä saa roskata luontoa.
Muista kirjoittaa kaikkiin vastauspapereihin nimesi ja opiskelijanumerosi.

Vastauksessa oleta vastaavasi sellaisen ihmisen esittämään kysymykseen, joka tuntee kohtalaisen hyvin ohjelmistotekniikan aihealuetta muutoin paitsi juuri tämän kysymyksen osalta. Mieti etukäteen vastauksesi pääkohdat ja lajittele ne johdonmukaiseen järjestykseen — älä kirjoita yhteen pötköön kaikkea mieleen tulevaa. Muista vastata kaikkiin tehtävän kysymyslauseisiin, sillä täysiä pisteitä ei voi saada jos kaikkiin kysyttyihin asioihin ei ole vastattu. Jos vastaus vaatii ohjelmakoodin kirjoitamista, sen ei tarvitse olla pilkulleen syntaksiltaan oikein. Mikä tahansa johdonmukaisesti käytetty ja yleisessä käytössä olevia ohjelmointirakenteita sisältävä koodin esitysmuoto käy.

Ohjelma 1: Vähennysohjelma

```
namespace moduuli {
  unsigned int CountDown = 42;

  void Vahenna() {
      unsigned int TyoKopio = CountDown;
      TyoKopio = TyoKopio - 1;
      CountDown = TyoKopio;
  }
}
```

1. Ohjelman 1 funktiota `Vahenna()` kutsuu ohjelman käynnistyksen jälkeen kaksi suoritussäiettä. Molempien tulisi vähentää muuttujan `CountDown` arvoa yhdellä, joten odotettu lopputulos on `40`. Näin ei ohjelmoijan hämmästykseksi kuitenkaan aina tapahdu. *6p.*

   (a) [3 pistettä] Miksi lopputulos voi ollä väärä? Selosta yksi suoritusketju, jossa odotettu lopputulos saadaan ja toinen, joka menee pieleen.

   (b) [3 pistettä] Mitä on suoritussäikeiden poissulkeminen? Miten sen avulla ohjelmassa voitaisiin varmistaa, että odotettu lopputulos saadaan laskennassa aina? Tee tarvittavat muutokset ohjelmaan.

2. Kerro lyhyesti mitä seuraavat asiat ovat? *11p.*

   (a) [1 piste] Rinnakkaisen ohjelman lukkiutuminen (deadlock)

   (b) [1 piste] Suoritussäikeen nälkiintyminen (starvation)

   (c) [1 piste] Suoritussäikeiden välinen synkronointi?

   (d) [2 pistettä] Atominen konekäsky. Anna esimerkki tällaisesta käskystä (selosta myös mitä käsky tekee – pelkkä nimi ei riitä)

   (e) [2 pistettä] rw-lukko (read-write lock). Miten toimii? Missä tilanteissa sitä tyypillisesti voi käyttää rinnakkaisessa ohjelmassa?

   (f) [2 pistettä] Minkälainen rinnakkaisuuden hallintamekanismi on futuuri (future) ohjelmointikielissä?

   (g) [2 pistettä] Mitä on ohjelmointikirjaston säieturvallisuus (thread safety)?

3. [6 pistettä] Minkälainen rinnakkaisuuden hallintamekanismi on Semafori? Semaforilla voi luonnin yhteydessä olla alkuarvona esimerkiksi nolla, yksi tai viisi. Mihin käyttötarkoitukseen kullakin alkuarvolla alustettua semaforia voi käyttää (mikä on oletettu käyttötarkoitus arvon perusteella)?

4. [6 pistettä] "Klassinen" rinnakkaisuuden esimerkki on tuottaja-kuluttaja (producer-consumer). Selosta minkälaisesta ohjelman rakenteesta on kyse? Minkälaisia rinnakkaisuuden ongelmia rakenteen käyttöön liittyy? Hahmottele säieturvallinen ohjelmakoodiratkaisu ongelmaan kun käytössäsi on PTHREADS-säiekirjaston operaatiot lukkojen (mutex) ja ehtomuuttujien (cond) käyttöön (säikeiden luonnin oletetaan tapahtuvan tämän koodin ulkopuolella, joten sitä ei tarvitse ottaa hahmotelmaan mukaan). Arvostelussa **ei** vähennetä pisteitä syntaksiltaan väärin kirjoitetusta ohjelmakoodista (kunhan merkityksen ymmärtää), eikä tarvitse tarkalleen muistaa `pthread_` rutiinien nimiä (kunhan merkinnästä ymmärtää mikä looginen operaatio tehdään.