TIE-02500 Rinnakkaisuus

Tentti 22.5.2015

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. [2 pistettä] Ohjelmakoodia suorittavilla suorityssäikeillä (threads) on paikallista ja yhteistä muistia. Mitä nämä termit tarkoittavat ja miten ne näkyvät ohjelmassa 1?

2. 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.

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

   (a) [1 piste] Suoritussäikeen vuoronnus (scheduling)

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

   (c) [1 piste] Mikä on aktiivinen olio (active object)?

   (d) [2 pistettä] Miten rinnakkaisuuden hallinta viestinvälityksellä (CSP tyyppiset ohjelmointikielet) eroaa säikeiden käytöstä?

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

   (f) [2 pistettä] Käänteisprioriteettiongelma (priority inversion problem)?

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

4. Rinnakkaisuutta tukevassa ohjelmointikielessä `coffeex` on tuki monitoreille. *7p.*

   (a) [1 piste] Minkälainen rakenne monitori on ohjelmoijan näkökulmasta?

   (b) [1 piste] Miten monitori pitää huolen poissulkemisesta?

   (c) [1 piste] Miten monitorin sisällä oleva säie voi odottaa toisen säikeen tehtävän valmistumista (synkronointi)?

   (d) [4 pistettä] Toisesta ohjelmointikielestä siirrettävä ohjelma käyttää semaforeja. Miten toteutat semaforin toiminnallisuuden monitorin avulla?

5. [4 pistettä] WWW-palveluun CatBook voivat käyttäjät ladata valokuvia lemmikeistään ja palvelu taltioi kuvat myöhemmin ihailtaviksi. Palvelu tekee tällä hetkellä seuraavat toimenpiteet luetellussa järjestyksessä:

   1. Lataa kuva käyttäjältä käsiteltäväksi (100 millisekuntia). Kehnosta verkko-ohjelmoinnista johtuen latausrutiin suoritus hidastuu, jos sitä kutsuu useampi säie kerrallaan (ensimmäinen säie 100 ms, jokainen seuraava säie hidastaa 1000 ms lisää kaikkia latauksia).
   2. Kerää kuvatiedoston metadatasta kuvauspaikan GPS-koordinaatit ja muu kiinnostava metainformaatio (10 ms)
   3. Muunna kuva mustavalkoiseksi (50 ms)
   4. Algoritmi etsii mustavalkokuvasta kaikki kissat (600 ms)
   5. Jokaista löydettyä kissaa varten tuotetaan uusi kuvatiedosto, johon on kopioitu alkuperäisestä kuvasta vain kyseinen eläin (100 ms per löydetty kissa)
   6. Alkuperäinen kuva, kerätty metainformaatio ja kaikki tuotetut uudet kissakuvat talletetaan tietokantaan (100 ms, kaikki tiedot talletetaan yhtenä tietokantakutsuna)

   Koska kaikki edellä kuvatut operaatiot pitää suorittaa loppuun ennen kuin seuraavan käyttäjän kuva voidaan ottaa käsittelyyn, niin palvelu on osoittautunut sen suuren suosion seurauksena liian hitaaksi. Hahmottele tälle sarjalliselle ohjelmalle uusi rakenne, jossa sen nopeutta parannetaan käyttäen rinnakkaisuutta. Kaaviokuva ja selostus rakenteesta riittää — ohjelmakoodia ei tarvitse kirjoittaa.