TIE-02500 Rinnakkaisuus

Tentti 12.5.2017

Tentin vastuuhenkilö: jyke.savia@tut.fi
Laskimen käyttö on kiellettyä.
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.

1. Termi **Data Race** on määritelty: *6p.*

   - kaksi tai useampi säiettä yhden prosessin sisällä käsittelee samaa muistipaikkaa
   - vähintää yksi käsittelyoperaatioista on muistiin kirjoitus
   - säikeet eivät koordinoi toimintaansa mitenkään (esim. lukoilla)

   (a) [2 pistettä] Mitä säie ja prosessi tarkoittavat määrittelyssä?

   (b) [2 pistettä] Anna esimerkki tilanteesta, jossa säikeet voivat toimia väärin Data Race:n takia (koodi ja selostus virheestä)

   (c) [2 pistettä] Mikä on säieohjelmoinnin lukko ja miksi se ratkaisee Data Race -tilanteen?

2. Barrier-synkronointi. *6p.*

   (a) [2 pistettä] Mikä on Barrier-rakenteen käyttötarkoitus rinnakkaisessa ohjelmoinnissa?

   (b) [1 piste] Anna käytännön esimerkki tilanteesta, jossa Barrieria voi/kannattaa käyttää rinnakkaisessa ohjelmassa

   (c) [3 pistettä] Määrittele C++-luokka, joka toteuttaa Barrierin. Hahmottele luokan rajapinta (metodit), toteutus ja anna myös esimerkki siitä miten Barrier-luokkaasi käytetään. (Keskity luokkaan ja siitä tehdyn instanssin eli olion käyttämiseen. Säikeiden luontia tms. EI tarvitse olla vastauksessasi mukana.)

Miten

3. Ovatko seuraavat väittämät **oikein** vai **väärin**?
Jos väärin, niin kerro mitä väittämässä on pielessä (vaikka kohdan väite olisi väärin, niin siitä ei saa pisteitä jos perustelua ei ole olemassa). *5p.*

(a) [1 piste] Ohjelma 1 toteuttaa poissulkemisen kunhan säikeet muistavat aina ennen kriittistä aluetta kutsua rutiinia `enter()` ja poistuessaan rutiinia `exit()`.

```
1  bool varattuna = false;
2
3  void enter()
4  {
5      while( varattuna == true )
6          ;
7      varattuna = true;
8  }
9
10 void exit()
11 {
12     varattuna = false;
13 }
```

Ohjelma 1: Poissulkeminen

(b) [1 piste] Ohjelmoija ei voi mitenkään tietää ohjelman 2 muuttujista ovatko ne säikeille yhteistä muistia vai pelkästään säikeen omassa käytössä.

```
1  namespace {
2      int A = 42;
3
4      void funktio( double param ) {
5          int i = 0;
6          // ...
7      }
8  }
```

Ohjelma 2: Muisti

(c) [1 piste] Säieturvallinen ohjelmakirjasto lupaa, että kirjaston rutiineja (funktioita) voi kutsua useammasta säikeestä ilman lukituksia (kirjasto itse pitää huolen siitä, että rutiinit toimivat luvatulla tavalla riippumatta yhtäaikaisten kutsujen määrästä).

(d) [1 piste] Säikeitä käyttävässä ohjelmassa voi olla käytössä vain yksi lukko (mutex).

(e) [1 piste] Rinnakkaisessa ohjelmoinnissa monitori on rakenne, joka tutkii ovatko säikeet menneet jumiin (deadlock).

4. Kerro lyhyesti mitä seuraavat asiat ovat? *4p.*

(a) [2 pistettä] Mitä on GPU (Graphics Processing Unit) laskenta? Miksi sen voidaan sanoa olevan massivisen rinnakkaista?

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