Ohjelmointitekniikka java, kevät 2004 26.4.2004
Tarkistuskertomukset tehtäville 3,5 ja 6.
Teemu Sjöblom

Tehtävä 3

Selitä millainen on ns. Strategia -suunnittelumalli, kun se tulkitaan kooditason arkkitehtuurin malliksi. Millaiseen tilantenteeseen mallia voidaan luontevasti soveltaa? Anna pieni Java-esimerkki. (Vastauksen maksimikoko on kaksi sivua.)

Suurin osa pisteistä tässä tehtävässä meni siihen, että vastauskessa kerrotiin yleensä Strategia-suunnittelumallista ja koska se on hyödyllinen, eikä siitä kuinka se kooditasolla toteutetaan. Koodiesimerkki osattiin antaa oikein useasti.

Tehtävästä sai pisteitä seuraavasti:

1p = Selitystä siitä, että kooditasolla meillä on rajapinta, sen toteuttavat luokat ja asiakas. Abstraktia luokkaa hyödyntävä ratkaisu kelpasi myös.
1p = Asiakkaalla on olioviite rajapinnan toteuttavaan luokkaan, minkä se voi vaihtaa lennosta toiseen saman rajapinnan toteuttavaan olioon ja tätä kautta vaihtaa käyttämäänsä/tarjoamaansa toiminnallisuutta.
1p = Kuvaus siitä millaiseen tilanteeseen tuo sopii, eli sellaiseen missä pitää vaihtaa toiminnallisuutta lennosta.
2p = Java-koodiesimerkki.

Tehtävä 5

Selitä säikeen tilat NEW, RUNNABLE, BLOCKED ja DEAD.
Miten näihin tiloihin joudutaan, ja miten niistä päästään pois?
Vanhentuneista (depricated) operaatioista ei tarvitse kertoa.

Tehtävä tarksitetiin siten, että jokaisesta vastauksesta etsittiin selitystä kaikille 12 yksittäiselle kysymykselle jotka tehtävänannossa on. Vastausten tulkinnassa oltiin opiskelijan puolella ja vastauksia luettiinkin tyyliin “jos se kerta osasi vastata tuon ja tuon kyllä se tämänkin osasi”. Esimerkiksi, jos opiskelija selitti kuinka muista tiloista pääsee pois, mainiten vaihtoehdot onnistuneesti, ei opiskelijalta rokotettu pisteitä siitä, että ei maininnut sitä, kuinka BLOCKED tilaan päästään. Tilasiirtymien selitykseksi hyväksyttiin myös tilakaavio, missä siirtymien nuolissa oli lyhyt kuvaus siirtymän aiheuttajasta. Koska vastaustilaa oli käytössä yksi sivu, ei vastauksesta vaadittu suuria yksityiskohtien tietämistä, pääpiirteiden selittäminen kelpasi säikeen elämästä kelpasi. Vastauksen kokorajan ylittämisestä ei sakoitettu.

Pisteitä tuli oikeista vastauksista seuraavasti:

>9 = 5 pistettä.
>7 = 4 pistettä.
>5 = 3 pistettä
>3 = 2 pistettä.
>0 = 1 piste.
=0 = 0!

Suurimpana huomiona vastauksista voisi mainita sen, että monet opiskelijat olivat jättäneet selittämättä tilat ja mitä tila tarkoittaa. Muutenkin kaikkiin tehtävänannoin kusymyksiin oli harvoin vastattu, 90% oli jättänyt vastaamatta kysymyksiin “kuinka DEAD tilasta pääsee pois ja mitä DEAD tila tarkoittaa”

Mitä opiskelijoilta haluttiin saada?

NEW
Tilaan joudutaan: Kun säie luodaan new Thread() operaatiolla.
Tila tarkoittaa: Säie olio on olemassa, systeemiresursseja ei ole, ainoastaan run()-metodia voi kutsua ilman poikekuksen syntymistä.
Tilasta poistutaan: run()-metodia kutsumalla.

RUNNABLE
Tilaan joudutaan: Threadi siirtyy RUNNABLE tilaan kun sen run() metodia kutsutaan, tai kun BLOCKED tilan aiheuttanut este on poistunut.
Tila tarkoittaa: Että säikeelle on jaettu systeemiresurssit käyttöön, se odottaa vain pääsyä suoritukseen.
Tilasta poistutaan: BLOCKED tilaan mikäli suoritukselle tulee este, esimerkiksi odotetaan I/O operaation valmistumista. DEAD tilaan, mikäli säikeen run()-metodin suoritus päättyy.

BLOCKED
Tilaan joudutaan: Threadin sleep()-metodia on kutsuttu, Threadi on kutsunut wait()-metodia ja odottaa tietyn ehdon täyttymistä, Threadi odottaa I/O:n päättymistä.
Tila tarkoittaa: Ei ole jonottamassa prosessoriaikaa vaan “paitsiossa”.
Pääse pois: Kun I/O valmistuu, sleep()-metodin aika on kulunut, odotettava este on väistynyt.

DEAD
Tilaan joudutaan: Kun run() metodin suoritus päättyy.
Tila tarkoittaa: Säikeen suoritus on päättynyt, sitä ei voida käynnistää uusiksi.
Pääsee pois: Tilasta ei pääse pois, roskienkerääjä syö olion kunhan kerkiää.

Tehtävä 6

Seuraava ohjelmahahmo ("idiomi") on tyypillinen:

```
void run () {
   try {
     while (! interrupted () && hasMoreWork)
       ... // may block and throw..
   }
   catch (InterruptedException e) ...
     // interrupted during sleep or wait
     ...
   finally
     ... // cleanup in any case
 }
```

Mistä on kysymys? (Vastauksen maksimikoko on yksi sivu.)

Tehtävässä pisteet tuli seuraavien asioiden mainitsemisesta vastauksessa.

1p = Kyseessä säikeen run() metodi.
1p = Säie tekee työtä kunnes sitä ei ole tai..
1p = ..tarkistaa aina interrupted() metodilla onko jokin muu säie pyytänyt suorituksen keskeytystä ja jos on, lopettaa suorituksensa.
1p = Catch lohko on olemassa koska voi tulla InterruptedException kun säie on sleep/wait tilassa.
1p = finally lohkossa siivotaan jäljet aina.

Tehtävä meni kaikilta erittäin hyvin.