81141 Ohjelmointikielten periaatteet
Tentti 1.3.2001

1. Virheitä, joita voidaan paljastaa, paljastetaan (1) alkioanalyysissä, (2) syntaksianalyysissä, (3) käännettyä ohjelmaa generoitaessa ja (4) käännettyä ohjelmaa ajettaessa. Voidaanko seuraavat virheet paljastaa tavanomaisen tyyppistä, mutta turvalliseksi suunniteltua kieltä käytettässä (esim. Adassa), ja jos voidaan, niin missä vaiheessa:

   a) käytetään tunnistetta, jota ei ole määritelty,
   b) sijoituksen kohde on väärää tyyppiä,
   c) jaetaan nollalla,
   d) aliohjelman parametrien määrä on virheellinen,
   e) taulukkoindeksi osoittaa ohi indeksirajojen,
   f) case-rakenteessa käytetään vaihtoehtoa, jota ei ole määritelty.

2. Mitä tarkoitetaan union-tyypillä? Miten se suhtautuu ns. tietuevariantteihin? Mitä eroa on vapaalla ja diskriminoidulla unionilla? Voiko tällainen tyyppi olla turvallinen?

3. Turvallisessa kielessä on osoittimiin kiinnitettävä erityistä huomiota. Kommentoi lyhyesti seuraavia väitteitä:

   a) Osoitinten käännösaikaiset tyyppitarkistukset ovat tehottomia, koska osoittimen osoittaman tiedon tyyppiä ei voi käännösaikana tarkistaa.
   b) On mahdotonta paljastaa käännösvaiheessa sitä, että käytetään vain osoittimia, jotka joko osoittavat jonnekin tai ovat null-osoittimia.
   c) Hyvin suunnitellussa kielessä null-osoittimien käyttö tarkistetaan jo käännösaikana.
   d) Null-osoittimet ja jäänneviitteet (dangling reference) ovat ajoaikana keskenään samankaltaisia ongelmia.
   e) Jäänneviitteitten käyttö estetään siten, että heap-muuttujaa deallokoitaessa siihen osoittavat osoittimet asetetaan null-osoittimiksi.
   f) Heap-muuttujien roskaantuminen voidaan estää viitelaskureilla (reference counter).

4. Aliohjelmassa P on sisempiä lohkoja, joissa määriteltyjen taulukkomuuttujien koot määräytyvät vasta aliohjelmaa suoritettaessa. Miten tapahtuu muistin allokointi näille taulukoille, ja miten käännetty ohjelma pystyy käyttämään niiden alkioita?