OHJ-1150 Ohjelmointi II – tentti 08.02.2010

tentin laatinut: Ari Suntioinen *<ari.suntioinen@tut.fi>*

Tentissä saa olla esillä kurssien OHJ-1100 Ohjelmointi I- ja OHJ-1150 Ohjelmointi II -luentomonisteet ja kurssin kotisivulta ladattava c++ -kirjastoreferenssi, joissa saa olla käsin tehtyjä lisämerkintöjä. Muuta lisämateriaalia tai laskimia ei saa olla esillä.

Tehtävä 1

Vastaa ***ensimmäisenä*** tämän tehtävän kysymyksiin:

(a) Kirjoita nimesi ja opiskelijanumerosi ***selkeästi*** jokaisen palauttamasi paperin etusivun oikeaan yläkulmaan. [1 p]

(b) Kopioi seuraava taulukko siististi päällimmäiselle vastauspaperille nimesi ja opiskelijanumerosi alle siten, että jokainen "ruutu" on kooltaan 2×2 konseptipaperin ruutua:

1	2	3	4	5	Σ

[1 p]

Tehtävä 2

Selitä lyhyesti (max. 3–4 virkettä) seuraavat:

(a) tiedon eheys, [5 p]

(b) staattinen muuttuja, [5 p]

(c) rakentajan eksplisiittinen kutsuminen, [5 p]

(d) muistipaikan ja osoittimen välinen yhteys ja [5 p]

(e) toiminnallinen abstraktio. [5 p]

Huomioi vastatessasi seuraavat:

– Esimerkki ei yksinään riitä vastaukseksi: anna yleinen selitys.

– Älä selitä kysyttyä termiä sen itsensä (tai sen taivutusmuotojen) avulla.

– Selitä yksikäsitteisesti: jos vastauksesi voi tulkita väärin, se tulkitaan väärin.

– Älä kopioi tekstiä sana–sanalta monisteesta: se ei osoita asian ymmärrystä (tähän tehtävään kannattaa yrittää vastata ensin kurkkimatta monistetta).

Huomaa myös, että kaikkiin kohtiin ei välttämättä löydy sanatarkkaa selitystä monisteesta, vaan joudut kertomaan, miten termin merkitys on sinulle avautunut.

Tehtävä 3

Seuraavassa on määritelty rekursiivinen funktio `func`, joka on kommentoitu niin huonosti, että kommenteista ei voi päätellä sen toimintaa.

```
void func(vector< vector<int> >& v, int i = 0, int j = 0) {
    if ( i >= v.size() ) {
        return;
    } else if ( j == v.at(i).size() ) {
        if ( ! v.at(i).empty() ) {
            v.at(i).back() = v.at(i).back() / 2;
        }
        func(v, i + 1);
    } else {
        if ( j == 0 ) {
            v.at(i).push_back(0);
        }
        v.at(i).back() = v.at(i).back() + v.at(i).at(j);
        func(v, i, j + 1);
    }
}
```

Tutki funktiota ja vastaa seuraaviin kysymyksiin:

(a) Mitä parametrivektorissa `v` on funktiosta palattaessa, jos vektori oli alunperin tyhjä? [2 p]

(b) Entä jos vektorissa oli alunperin yksi alkio? [3 p]

(c) Mitä funktio yleisesti ilmaistuna tekee parametrivektorilleen? [8 p]

(d) Onko rekursiivinen funktio, jonka paluuarvo on tyypiltään `void`, aina automaattisesti häntärekursiivinen? [5 p]

(e) Toteuta ilman rekursiota funktion `func` kanssa samoin käyttäytyvä funktio. [7 p]

Käännä ➠

Tehtävä 4

Ajatellaan seuraavanlaisista dynaamisesti varatuista Node-structeista:

```
struct Item {
    Item *next;
    int alkioiden_summa;
};
```

koostuvaa linkitettyä rakennetta, jonka sijainnista muistissa pidetään kirjaa osoittimella `lisatty`:

```
Item *lisatty = NULL;
```

Rakenteelle on määritelty lisäysalgoritmi (tilanteen yksinkertaistamiseksi koodissa on oletettu, ettei muistin varaus epäonnistu):

```
Item *varattu = new Item;
varattu->alkioiden_summa = summattava;
if ( lisatty != NULL ) {
    varattu->next = lisatty->next;
    lisatty->next = varattu;
} else {
    varattu->next = varattu;
}
lisatty = varattu;
```

Nämä määrittelyt huomioiden:

(a) Kuvaile sanallisesti mahdollisimman yleisellä tasolla, mihin rakenteen alkioon `lisatty` osoittaa. [1 p]

(b) Kuvaile sanallisesti mahdollisimman yleisellä tasolla, mihin rakenteen alkioon `lisatty->next` osoittaa. [4 p]

(c) Piirrä laatikko–nuoli-kaaviona tilanne, kun tyhjään rakenteeseen on lisätty järjestyksessä alkiot 1, 3 ja 2. [6 p]

(d) Esitä ylläolevan kaltaisena c++-algoritmina, kuinka poistaisit rakenteesta siellä kauimmin olleen alkion. [6 p]

(e) Esitä c++-algoritmina, kuinka tulostaisit rakenteeseen tallennetut alkiot siinä järjestyksessä, kun ne on sinne lisätty. [8 p]

Tehtävä 5

Suunnittele ***STL-säiliöiden avulla*** tietorakenne/tietorakenteet, johon/joihin voitaisiin tallentaa ennalta määräämätön määrä tietoa ihmissuhteista: siis tietoja siitä, tunteeko henkilö A henkilön B. Sovitaan lisäksi, että "tunteminen" on kaksisuuntainen ilmiö: jos A tuntee B:n, tällöin automaattisesti myös B tuntee A:n.

(a) Esitä määrittely muuttujalle/muuttujille, johon/joihin ihmissuhdeverkko voitaisiin tallentaa. [8 p]

(b) Esitä c++-käsky tai -käskyt, joilla rakenteeseen lisättäisiin tieto: *Matti tuntee Maijan.* [2 p]

(c) Esitä c++-käsky tai -käskyt, joilla rakenteesta saataisiin selville *Tunteeko Jaakko Jaanan?* [3 p]

(d) Esitä c++-käsky tai -käskyt, joilla rakenteesta saataisiin selville *Kaikki Minnan tuntemat henkilöt.* [4 p]

(e) Esitä c++-käsky tai -käskyt, joilla rakenteesta saataisiin selville *Onko Terolla ja Kallella yhteisiä tuttavia?* [8 p]

Pisteitä saa paremmin, jos toteutettu rakenne on sellainen, että operaatioissa ei tarvita lineaarista hakua yms., vaan ne on mahdollisuuksien mukaan toteutettu suoraan käytettyjen STL-säiliöiden omilla operaatioilla. Edellä ilmaisulla "saataisiin selville" ajetaan takaa sitä, että tiedot tulevat operaation toteuttavalta funktiolta jotenkin takaisin kutsujalle. Ei sitä että ne tulostetaan näytölle.