Ohjelmoinnin edistyneet piirteet – OSA II – Harjoitukset 4

Harjoitusten opittavat aihealueet

- Kopiorakentaja
- Olioiden välittäminen funktioiden/metodien parametrina
- Const –sanan käyttö funktioiden/metodien parametrien yhteydessä
- Pieni johdatus STL –tietorakentesiin (Standard Template Library) ja olioiden säilöminen dynaamiseen vector –tietorakenteeseen.
- Osoittimien vs. olioiden tallettaminen vector –tietorakenteeseen.

Tehtävä 1 (1p). Kopiorakentaja. Luo edellisen viikon Kalenterimerkintä –oliolle kopiorakentaja, jossa kopioit vain tietokentät ”asia” ja ”muistutus”. Älä kopioi kalenterimerkinnän päiväystä, vaan kysy kopiorakentajassa olion päiväys –kentän tiedot käyttäjältä ja aseta päiväys käyttäjän antamien tietojen pohjalta (päivä, kuukausi ja vuosi).

Tehtävä 2 (1p). Olioiden välittäminen funktioille/metodeille sekä const –sanan käyttö funktioiden/metodien viittausparametrien yhteydessä. Luo kaksi yksinkertaista funktiota/aliohjelmaa, joille välität Kalenterimerkintä –olion parametrina. Toisessa aliohjelmassa välitä aliohjelmalle kalenterimerkintä tavallisena arvoparametrina ja toisessa viittausparametrina. Havainnoi kopiorakentajan käyttö.

```
void doSomethingArvoparametri( Kalenterimerkinta aKalenterimerkinta ) …

void doSomethingViittausparametri( Kalenterimerkinta& aKalenterimerkinta ) …
```

Miksi ensimmäistä versiota ei tulisi lähtökohtaisesti C++ -kielessä käyttää (olioparametrin välitys arvoparametrina)?

Muuta viittausparametri nyt const –tyyppiseksi.

```
void doSomethingViittausparametri( const Kalenterimerkinta& aKalenterimerkinta ) …
```

Mitä parametrina annetun olion metodeja pystyt nyt kutsumaan ja miksi? Milloin const –sana tulisi laittaa parametriviitteen yhteyteen?

C++ -käytäntö: Muuta nyt myös kaikki sellaiset luokkien metodien parametrit (mm. setterit) const –viitteeksi, joissa annat olion parametrina ja joissa ei muuteta parametrina annetun olion sisäistä tilaa (eli käytetään vain olion const –metodeja). Tämä käytäntö estää turhat ja raskaat olioiden kopioinnit metodikutsuissa.

Tehtävä 3 (2p). Pieni henkilötietoja tallentava sovellus. Kirjoita ohjelma, jossa on silmukassa 3 toimintoa:

1. Lisää uusi henkilö
2. Tulosta henkilöt
3. Lopeta

Lisää uusi henkilö –toiminto kysyy henkilön nimen, iän ja osoitteen, minkä jälkeen luodaan tietojen pohjalta uusi Henkilö –olio, joka lisätään ohjelman alussa luotuun Henkilö –vektoritietorakenteeseen. Tietorakenteen voit luoda heti pääohjelman alussa esim:

vector<Henkilo> henkilot;

```
henkilot.push_back( Henkilo(”Anne”, 20) );

henkilot.push_back( Henkilo(”Kalle”, 30 ) ); // jne
```

Tulosta henkilöt –toiminto tulostaa koko tietorakenteen sisällön eli kaikki henkilöt konsoliin.

Lopeta lopettaa ohjelman.

Tehtävä 4 (1p) Vektorillinen olioita vai vektorillinen osoittimia olioihin

Tärkeää ymmärtää C++ -kielestä: Tehokkuusnäkökohtia ja pohdintaa. Luo ohjelmassasi 5 henkilöä ja lisää ne vektoriin. Muuta vektorin sisältöä siten, että talletat vektoriin Henkilö –osoittimia Henkilö –olioiden sijaan. Esimerkkikoodia:

```
vector<Henkilo*> henkilot;

henkilot.push_back( new Henkilo(”Anne”, 20) );

henkilot.push_back( new Henkilo(”Kalle”, 30 ) ); // jne
```

Varmista –että sinulla on Henkilö –luokkaan toteutettu tehtävän mukainen kopiorakentaja, jossa tulostat tyyliin ”Henkilo –luokan kopiorakentaja” ja purkaja tyyliin ”Henkilo” –luokan purkaja.

Vertaile tilannetta, jossa vektorissa on Henkilö –olioita osoittimien sijaan. Montako kertaa rakentajaa/kopiorakentajaa kutsutaan lisättäessä vektoriin 5 henkilöä. Kumpi on tehokkaampaa ja miksi näin on?

- ⇨ C++ -kielessä tulisi yleensä melkein aina tallettaa tietorakenteisiin osoittimia itse olioihin eikä itse olioita. Näin toimivat sisäisesti muutkin oliokielet (esim. Javassa vektori sisältää aina viittauksia/osoittimia olioihin, vaikka näyttää, että tietorakenteeseen talletettaisiin olioita.

Vapaaehtoinen lisähaaste tehtävään 3. 😊

Toteuta ohjelmaan myös toiminto 4. Poista henkilö. Toiminnossa ohjelma kysyy poistettavan henkilön nimeä, minkä jälkeen ohjelma poistaa kyseisen henkilön tiedot vektorista, jos henkilö löytyy.