OHJ-1101 Ohjelmointi Ie, tentti

1.2.2008, Essi Lahtinen

Kirjoita jokaiseen vastauspaperiisi selkeästi ylläolevat otsikkotiedot, nimesi, opiskelijanumerosi, koulutusohjelmasi, vuosikurssisi, vastauspaperin järjestysnumero ja jättämiesi vastauspapereiden kokonaismäärä. Jos olet korottamassa aikasempaa suoritustasi, kerro, miltä vuodelta suorituksesi ovat. *Jätä vastauspapereihisi vähintään 3cm ulkoreunamarginaalit ja ensimmäiseen paperiin vähintään 10 cm yläreunamarginaali.* Kirjoita selkeällä käsialalla. Tentin tarkastaja ei ole hieroglyfien erikoisasiantuntija. Onnea tenttiin!

Tehtävä 1

1. Mitä hyötyä seuraavasta määrittelystä on? `const int KOKO = 10;` (1p.)
2. Oletetaan, että tarvittavat include-direktiivit jne. ovat käytössä ja määriteltynä on muuttuja `int luvut[ KOKO ]`. Kuinka monta sijoitusta seuraavissa ohjelmapätkissä suoritetaan (A ja B erikseen)? (1p.)

```
// Ohjelmapätkä A
int tmp = luvut[ 0 ];
for( int i = 0; i < KOKO-1; ++i ) {
   luvut[ i ] = luvut[ i+1 ];
}
luvut[ KOKO-1 ] = tmp;
```

```
// Ohjelmapätkä B
for( int i = 0; i < KOKO - 1; ++i ) {
   int tmp = luvut[ i ];
   luvut[ i ] = luvut[ i+1 ];
   luvut[ i+1 ] = tmp;
}
```

3. Selitä yksinkertaisesti suomeksi, minkä *toimenpiteen* kohdassa 2 esitetyt ohjelmapätkät suorittavat taulukolle (A ja B erikseen)? (2p.)
4. Vertaile kohdan 2 ohjelmapätkän A ja ohjelmapätkän B toteutusta keskenään. Kumpi on järkevämmin toteutettu? Perustele. (2p.)

Tehtävä 2

1. Mitä ovat `Juttu` ja `j` seuraavissa määrittelyissä? (1p.)

```
struct Juttu { int a; int b; };
Juttu j;
```

2. Määriteltynä on edellisessä kohdassa määriteltyjen lisäksi muuttuja `Juttu jutut[ KOKO ]`. Mitä vikaa seuraavissa kahdessa ohjelmarivissä on? (2p.)

```
jutut.a = 42;
cout << j << endl;
```

3. Tarvitaan ohjelma, joka tallettaa tietoja F1-kisasta: Jokaisella kuskilla on oma kilpailunumero, jonka perustella pitää saada selville kuskin nimi ja jokaisen kierroksen kierrosaika sekä se, millaisilla renkailla ko. kierros on ajettu (pehmeillä vai kovilla). Minkälaista C++-rakennetta käyttäisit tietojen tallentamiseen? (4p.)
4. Kirjoita ohjelmakoodipätkä, joka etsii em. tietorakenteesta kilpailija Kimi Räikkösen kolmannen kierroksen kierrosajan, kun Kimin kilpailunumeroa ei tiedetä. (3p.)

Tehtävä 3

1. Oletetaan, että tarvittavat include-direktiivit jne. ovat käytössä. Mitä seuraava ohjelmapätkä tulostaa? (3p.)

```
bool foo( int i, int& j ) {
   ++i;
   --j;
   if( i + j > 80 ) {
      cout << "i " << i << endl;
      return true;
   }
   cout << "j " << j << endl;
   return false;
}

int main() {
   int luku = 42;
   while( foo( luku, luku ) ) {
      cout << "main " << luku << endl;
   }
   return EXIT_SUCCESS;
}
```

2. Mediaani eli keskiluku on järjestetyn joukon keskimmäinen alkio. Jos alkioiden määrä on parillinen, mediaaniksi lasketaan usein kahden keskimmäisen luvun keskiarvo. Esimerkki: Joukon { 2, 2, 3, 8, 14 } mediaani on 3 ja joukon { 2, 2, 3, 8 } 2,5.

   - Kirjoita C++:aa käyttäen esittely yleiskäyttöiselle funktiolle, joka ottaa kutsujaltaan joukon lukuja ja ilmoittaa kutsujalle ko. joukon mediaanin. (2p.)
   - Kirjoita määrittely edellisessä kohdassa esitellylle funktiolle. (2p.)
   - Kirjoita testipääohjelma, joka kutsuu edellä määriteltyä funktiota laskeakseen mediaanin jollekin joukolle lukuja ja tulostaa laskennan tuloksen tai virheilmoituksen, mikäli mediaania ei voitu laskea esim. siitä syystä, että lukujoukko ei ollut järjestyksessä tai joukko oli tyhjä. (2p.)

3. Nappulahyppelypelissä on yksi pelaaja. Idea on, että neliönmuotoinen lauta on aluksi muuten täynnä nappuloita, mutta keskellä on tyhjä ruutu. Nappulalla voi hypätä yhden nappulan sisältämän ruudun yli tyhjään ruutuun, jolloin ruudusta, jonka yli hypättiin poistetaan nappula. Tavoitteena on, että jäljelle jää vain yksi nappula. Peli halutaan toteuttaa C++-ohjelmana. Suunnittele ohjelman funktiojako. Kerro jokaisesta funktiosta sen *kuvaava nimi* ja n. yhdellä lauseella, mitä se tekee. (3p.)

4. Onko seuraava funktio toteutettu hyvin? Miten sen voisi toteuttaa järkevämmin? Perustele. (2p.)

```
// Tarkastaa kumpi vectoreista sisältää enemmän positiivisia alkioita.
// Parametrit: tarkasteltavat vectorit.
// Paluuarvo: < 0, jos v1:ssä enemmän; == 0, jos yhtäpaljon; > 0, jos v2:ssa enemmän
// Toiminta: Käy vectorit läpi ja vertailee jokaista alkiota vuorollaan.
// Ei sivuvaikutuksia.
int kummassaEnemmanPositiivisia( vector< int > v1, vector< int > v2 );
```