Jatkolio, bitin tentti 20.5.2013

1. Selitä seuraavat termit

   a) Funktio-olio
   b) Pysyvyys- ja vaihtelevuusanalyysi
   c) Poikkeushierarkia
   d) Viipaloituminen
   e)
   f) Moniperiytyminen

2.

   a) Mitä seuraavat asiat tarkoittavat ja mitä ne erikseen merkitsevät kutsujan ja kutsuttavan kannalta?

   i. Esiehto
   ii. Jälkiehto
   iii. Invariantti

   b) Määritä seuraaville metodeille esi- ja jälkiehdot mahdollisimman fiksusti ja niin tarkasti kuin osaat. Voit määritellä ne selittämällä tai kaavoin. Seuraavissa ohjelmissa on käytössä globaali muuttuja (hyi) std::vector<int> v.

   i. `double f1( double x, double y ) { return x/y; }`
   ii. `unsigned int f1( string s ) { return s.length(); }`
   iii. `int f3( int i ) { return v.at( i-1 ); }`
   iv. `int f4( int i ) { return v[ i+1 ]; }`
   v. `void f5() { v.push_back( v[0] }`

3. Selitä, miten seuraavat termit liittyvät toisiinsa olio-ohjelmoinnissa. Huomioi, että yhtymäkohtia saattaa olla useampia kuin yksi.

   a) Periytyminen - Olioiden kopioiminen
   b)
   c) Sopimussuunnittelu - Poikkeukset
   d) Osoittimet - Olioiden omistusvastuu/elinkaari
   e) Rajapintaluokat - Periytymishierarkia
   f) Pysyvyys- ja vaihtelevuusanalyysi - Templatet

4.

   a) Mitä ovat eri poikkeustakuut, mistä niissä on kyse ja miten ne hyödyttävät luotettavien ohjelmien suunnittelemista?

   b) Mitä poikkeustakuita seuraavan ohjelmalistauksen eri jäsenfunktiot tarjoavat? Perustele vastauksesi hyvin. Tässä tehtävässä saa olettaa, etteivät vectorin size(), erase() ja begin() heitä poikkeuksia ja että at() ja push_back() jättävät poikkeuksen heitettyään vectorin ennalleen.

```
#include <vector>

class Keke
{
    public:
        Keke() : data_(0)
        {}
        ~Keke();

        void lisaa( int x )
        {
            data_.push_back( x );
        }

        int montakoLoytyy( int x ) const
        {
            int lkm = 0;
            for( unsigned int i = 0; i < data_.size(); ++i )
            {
                if( data_.at( i ) == x )
                {
                    ++lkm;
                }
            }
            return lkm;
        }

        int vaihdaEkaJaVika()
        {
            int tmp = data_.at( 0 );
            data_.erase( data_.begin() );
            data_.push_back( tmp );
        }

        int monistaVikaNKertaa( int n )
        {
            int vika = data_.at( data_.size() - 1 );
            for( int i = 0; i < n; ++i ]
            {
                data_.push_back( vika );
            }
        }

        int tuplaaVikaJosSeOnUniikki()
        {
            int vika = data_.at( data_.size() - 1 );
            if( montakoLoytyy( vika ) == 1 )
            {
                monistaVikaNKertaa( 1 );
            }
        }

    private:
        std::vector<int> data_;
}
```