Examen d'Objet Avancé – ZZ3F2 – 2011

Durée : 2h – Documents non autorisés – Barème donné à titre indicatif, peut être sujet à modifications

Implémentation des concepts objets

1 – Donnez et définissez les trois grands principes de la programmation orientée objet. (1.5 pt)

2 – Qu'est-ce que le « name mangling » ? A quoi cela sert-il ? (1 pt)

3 – Par quel mécanisme le polymorphisme est-il possible en C++ (et dans la majorité des langages orientés objets) ? Décrivez ce mécanisme (vous pouvez illustrer votre explication par des schémas et/ou du code si vous en ressentez le besoin). (3 pts)

4 – En cours, nous nous sommes limités à l'implémentation de l'héritage simple. On désire maintenant supporter l'héritage multiple. Quels problèmes cela peut-il engendrer ? Comment résoudriez-vous ces problèmes (je n'attends pas de code, juste des idées générales) ? (3 pts)

Optimisation

1 – Qu'est-ce que le « loop unrolling » (ou « loop unwinding ») ? (1 pt)

2 – Optimisez la fonction suivante pour (potentiellement) diminuer le temps d'exécution, en justifiant vos choix. (3 pts)

```
void logErrors( std::vector< Error > errors )
{
    bool isInteractive = application.isInteractive();

    date d = getCurrentDate();

    std::vector< Error >::const_iterator it = errors.begin();

    for ( ; it != errors.end() ; ++it )
    {
        logToFile( date.toString() + ":" + it->message() );

        if ( isInteractive )
        {
            logToScreen( date.toString() + ":" + it->message() );
        }
    }
}
```

[Unnamed]

Vous travaillez sur une application quelconque. Cette application manipule des données qui sont sauvegardées dans un fichier. L'application est multi-threadée ; certains threads lisent le fichier, d'autres y écrivent des données. Par conséquent, pour éviter qu'un thread n'essaye de lire un fichier alors qu'un autre est en train de le modifier, tous les accès au fichier sont protégés par un mutex (classe `Mutex`, possédant une méthode `lock` et `unlock`). La sauvegarde est effectuée par un `Serializer`, qui sera soit un `XmlSerializer`, soit un `BinarySerializer` (selon le paramètre fourni). La classe `Serializer` possède un constructeur prenant en paramètre un flux, et une méthode `serialize` qui sérialise l'objet passé en paramètre. Voici un extrait du code permettant d'effectuer la sauvegarde des données :

```
class Data
{
    ...
    void save( const std::string & fileName, OutputType outputType )
    {
        mutex_.lock();

        std::ofstream outputFile( fileName );

        Serializer * s = NULL;
        if ( outputType == XML ) s = new XmlSerializer( outputFile );
        else                     s = new BinarySerializer( outputFile );

        s.serialize( * this );

        delete s;

        mutex_.unlock();
    }

    static Mutex mutex_;
};
```

Quelques jours après la mise en production, plusieurs tickets apparaissent dans le bug tracker indiquant que de temps en temps, l'application « freeze », elle ne répond plus, et les utilisateurs sont obligés de tuer le processus.

1 – Quelle est selon vous la cause de ce comportement ? (1 pt)

2 – Proposez une solution pour résoudre ce bug. (2 pt)

Boost

1 – Observez les fonctions suivantes :

```
std::string repeat( std::string const & exp, unsigned int times )
{
    return "(?:" + exp + "){" + boost::lexical_cast< std::string >( times ) + "}";
}

bool isValidPort( std::string const & portStr )
{
    // 1
    std::string const decimalByte = "(?:"
                                        "[1-9]?[0-9]|"
                                        "1[0-9][0-9]|"
                                        "2[0-4][0-9]|"
                                        "25[0-5]"
                                    ")";

    // 2
    std::string const ipAddress = repeat( decimalByte + "\\.", 3 ) + decimalByte;

    // 3
    std::string const portNumber = "(?:"
                                       "[0-9]|"
                                       "[1-9][0-9]{1,3}|"
                                       "[1-5][0-9]{4}|"
                                       "6[0-4][0-9]{3}|"
                                       "65[0-4][0-9]{2}|"
                                       "655[0-2][0-9]|"
                                       "6553[0-5]"
                                   ")";

    // 4
    boost::regex exp( ipAddress + ":" + portNumber );

    // 5
    return boost::regex_match( portStr, exp );
}
```

a) Quel est le but de la fonction isValidPort ? Donnez un exemple d'utilisation. (1 pt)

b) Donnez les commentaires que vous insèreriez aux endroits indiqués par les chiffres 1 à 5. (1 pt)

2 – Expliquez le fonctionnement du constructeur de la classe ci-dessous : (1.5 pt)

```
struct Port
{
    Port( std::string const & portStr )
    {
        assert( isValidPort( portStr ) );

        std::vector< std::string > tokens;
        boost::split( tokens, portStr, boost::is_any_of( ":" ) );

        ip_         = tokens[ 0 ];
        portNumber_ = boost::lexical_cast< unsigned short >( tokens[ 1 ] );
    }

    std::string    ip()         const { return ip_; }
    unsigned short portNumber() const { return portNumber_; }

  private:
    std::string    ip_;
    unsigned short portNumber_;
};
```

3 – On dispose de la fonction suivante :

```
template < typename Range, typename OutputIterator >
void xxx( Range r, OutputIterator out, unsigned short num )
{
    using boost::phoenix::arg_names::arg1;

    boost::remove_copy_if( r, out, boost::phoenix::bind( & Port::portNumber, arg1 ) != num );
}
```

Quel est le but de cette fonction ? Donnez-lui un nom plus parlant que « xxx ». (1 pt)