Course C^{++}, Exercise Number 6

22.04.2012 + two weeks

Topic of the exercise is exceptions, and still a bit of inheritance. Consider:

```
struct arithmetic_error
{
   std::string reason;

   arithmetic_error( const std::string& reason )
      : reason( reason )
   { }

   virtual void print( std::ostream& ) const;

};


struct division_by_zero : public arithmetic_error
{
   double x; // Number that we tried to divide by zero.

   division_by_zero( double x )
      : arithmetic_error( "division by zero" ),
        x(x)
   { }

   virtual void print( std::ostream& ) const;
};


struct negative_sqrt: public arithmetic_error
{
   double x; // Number that we tried to take square root of.

   negative_sqrt( double x )
      : arithmetic_error( "tried to take root of negative number" );
        x(x)
```

```
   { }

   virtual void print( std::ostream& ) const;
};
```

1. Write the `print( std::ostream& ) const` functions. Write a function
   `std::ostream& operator << ( std::ostream&, const arithmetic_error& e );`
   (It can be written in three lines.)
2. Write a few arithmetic functions, for example, `mydivide, mysqrt, mylog, myexp,` that throw suitable exceptions. You can also add more exceptions if you want.

Test your code with a block of the following form:

```
try
{
   ... operations
}
catch( const arithmetic_error& err )
{
   std::cerr << "something terrible happened: ";
   std::cerr << err << "\n";
}
```

```
It may be necessary to provide destructors and copy constructors.
Just try it out, and add them if the compiler complains.
```