COMP 322 - Introduction to C++

Winter 2011

Test 2

TA Practice Exercises

Note:

These exercises were inspired by the following textbook:

Learning C++: A Hands-On Approach
Eric Nagler, 1997

1. What is 'overloading resolution'? Briefly explain how the compiler determines which overloaded function to call.

2. Which of the calls to **`f`** in the following code snippet are valid/invalid? Explain.

```
void f( int );
void f( char, int = 0 );
void f( float, int );
void f( int, double );
void g()
{
     f('A');
     f( 1, 'A' );
     f( 1, 3.4 );
     f( 3.4 );
     f( 'A', 3.4 );
     f( 3.4, 1 );
}
```

3. Which of the following code snippets works, and which doesn't? Explain.

```
void f( int );
void f( char* );
void g()
{
     f(0);
}


void f( long );
void f( char* );
void g()
{
     f(0);
}
```

4. a) Briefly explain what operator function overloading is, and give an example of how/where it could be used.

b) List the seven operators which cannot be overloaded.

5. a) In the code below, we are attempting to advance to the next day in `foo()`. Rewrite the code using operator overloading to make it valid.

```
enum days
{
     Sunday, Monday, Tuesday, Wednesday, Thursday,
     Friday, Saturday
};

void foo( days& d )
{
     ++d; // Error, invalid operation
}
```

b) Write a `main()` function that will result in the following output:

```
Today is Sunday
Today is Monday
Today is Tuesday
Today is Wednesday
Today is Thursday
Today is Friday
Today is Saturday
Today is Sunday
Today is Monday
Today is Tuesday
```

6. a) Define inheritance and polymorphism.

   b) Explain how you would define a singly inherited derived class.

7. Explain what happens upon compiling/running the following code snippets.

a)

```
class String
{
     public:
     String ( const char* = "" );
};

class DString : public String
{
     // Nothing new here
};

void foo()
{
     DString dtest( "Test" );
}
```

b)

```
class String
{
     public:
     String upper() const;
     char* ptr;
};

class DString : String
{
     public:
     DString ( const String& );
};

void foo( DString& d )
```

```
{
     DString dtest(d.upper());
}
```

8. Should a base class destructor always be declared `virtual`? Explain.

9. a) Explain how you would write a function template. Where should it be placed in the code?

b) Write a function `max()` that takes two generic types and returns the greater of its two input values. Include a `main()` with examples of how `max()` can be called.

10. What does the following code snippet output (if anything)? Explain.

```
# include <iostream.h>

template < class T >
inline const T& max( const T& x, const T& y )
{
     return (x > y) ? x : y;
}

int main()
{
     int a = 5;
     double b = 6.1;
     cout << max(a, b) << '\n';
     return 0;
}
```

11. Write a function called `quadratic()` that computes and returns one value of *x* in the quadratic equation $Ax^2 + Bx + C = 0$ (by finding the discriminant, etc.). The coefficients are its arguments. This function must check for the denominator equal to zero, and also for a negative discriminant, and throw exceptions when these cases occur. Also write a `main()` that takes in coefficient values from the user, and attempts to call `quadratic()` and output its result. The program should not crash!