2010.2 Object-Oriented Programming and Design
Midterm Exam (Oct. 26th 5pm-6:30pm)

supervisor signature	

StudentID# : (   ) , Name : (   )

* You may answer in either Korean or English.

1. (28 points) Complete following sentences by filling out the blanks (a)~(l) with the most appropriate words.

(1) (a.   ) is defined as modification of a software product after delivery to correct faults to improve performance, or to adapt the product to a modified environment. (a.   ) is considered to be the most (b.   ) part of the software life cycle.

(2) (c.   ) is the software feature to be able to reuse the existing code instead of creating new code when moving software from one environment (platform) to another.

(3) (d.   ) involves the splitting of complex program tasks into smaller pieces with clear flow-control and interfaces between components.

(4) The advantage of using data encapsulation comes when (e.   ) of the class changes but (f.   ) remains the same.

(5) (g.   ) contains a boolean expression that you believe will be true when (g.   ) executes.If it is not true, the system will throw an error. By verifying that the boolean expression is indeed true, the (g.   ) confirms your assumptions about the behavior of your program. Experience has shown that writing (g.   ) while programming is one of the quickest and most effective ways to detect (h.   ).

(6) A good software design recognizes the inevitability of (i.   ) and plans an accommodation for them from the very beginning.

(7) If you need to write a copy constructor, you also need to write (j.   ) and (k.   ).
Why? Explain in detail : (l.   )

2. (10 points)

(1) Generally, overusing global variables is not desirable. Why? Explain WITH FULL DETAILS.
(   )

(2) What is the main advantages of using object-oriented programming compared to using structured programming? Explain.
(   )

4. (8 points) Explain following concepts.

(1) Cohesion : (   )

(2) Coupling : (   )

(3) shallow copy : (   )

(4) deep copy : (   )

5. (10 points)

(1) It is inefficient to pass large objects by value. Why? Explain in detail.
(   )

(2) In C++ programming, there can be one beneficial effect (advantage) of using "pass-by-value". What is the beneficial effect? Explain. Your answer should include explanation on why it is a beneficial effect.
(   )

6. (28points) In Problem 2 of Project #2, class **inf_int** is defined as follows. It can represent an arbitrarily big integer number in practice and the limit it can represent is maximized.

In (1) and (2), Do not call other member functions. Instead, You may use standard library in your code.

Your code should manage the memory correctly and efficiently. Your code should be grammatically and logically correct.

```
class inf_int
{ private:
        char* digit;            // points to a string of digits. Perform dynamic allocation when necessary.
        unsigned int length;     // stores the number of actual digits
        bool thesign;           // we assume thesign is 1 if positive integer, and 0 if negative integer.
 public :
        inf_int(const inf_int& x);                                        // (1) copy constructor
        friend bool operator<(const inf_int& l , const inf_int& r); // (2) operator overloading
       // other member functions should be here but they are not shown.
};
```

---

(1) Write C++ code for the copy constructor.

```
inf_int::inf_int(const inf_int& x)
{
}
```

(2) Write C++ code for the **operator<** function.

```
bool operator<(const inf_int& l , const inf_int& r)
{
}
```

(3) In fact, **inf_int** type number has the limit it can represent. What is the maximum number it can represent? Justify your answer.

(a) possible maximum number : ( )

(b) Justification : ( )

※ The answer of (a) can be an exact number or it can be something you can explain.

7. (7 points) Generally, we can say that it is desirable to use "**friend**" function/class only when it is really necessary.

That means, the usage of "**friend**" function/class often needs to be minimized. Why? Explain in detail.

( )

8. (9 points) To answer to the question "What is good software?", List at least six different properties of good software, which are often used as criteria to measure the quality of software.

(a.) , (b.), (c.)
(d.) , (e.), (f.)