2014.2 Object-Oriented Programming and Design
Midterm Exam (Oct. 23th 7pm-8:10pm)

supervisor signature	

StudentID# : ( ) , Name : ( )

* You may answer in either Korean or English in problem 2~3. However, you should insert only English words in problem 1.

1. (20 points) Fill out the blanks (a)~(f) with the most appropriate English words.

- Fred Brooks states that over 90% of the costs of a typical system arise in the (a.) phase.
- If we did not declare any constructor in class, the compiler would generate (b.).
- (c.) is a simple dummy routine with no behavior or with very limited behavior, which can be used when components are slowly integrated into completed system..
- **const** member function does not (d.) any data members of the object on which it is called.
- One of the main benefits of data encapsulation is (e.) which means a change in (f.) should not affect interface seen by clients of a class.
- (g.) is the degree to which the tasks assigned to a component seem to form a meaningful unit.
- Coupling is the degree to which the ability to fulfill a certain responsibility (h.) upon the actions of another component.
- **friend** function of a class has the right to access (i.) members of the class.
- One of the main goals of software engineering is to minimize (j.) of software development and maintenance.

2. (18points) (1) What is inline function? explain with sufficient details.
( )

(2) <u>In what situations</u>, can we get benefits of using inline function? List at least two situations.
(1:)

(2:)

(3) What benefit can we obtain by using inline function? Why?
(What benefit?)

(Why?)

3. (12points)
(1) What is “shallow copy”? Explain.
( )

(2) You don’t need to write a copy constructor if shallow copies are OK. Why? Explain.
( )

4. (18points) Consider an integer stack class **TIntStack** (posted on our class webpage) that is defined as follows.

```
#include <iostream>
const unsigned int DEFAULT_SIZE = 256;

class TIntStack {
public :
    TIntStack(unsigned int stackSize = DEFAULT_SIZE);
    TIntStack(const TIntStack& that); // copy constructor
    TIntStack& operator=(const TIntStack& assign);
    ~TIntStack(); // Destructor
    void Push(int thisValue);
    int Pop();
    unsigned int HowMany() const;

private :
    int* _sp; // memory is dynamically allocated
              // during constructor function call
    unsigned int _count; // number of elements in stack
    unsigned int _size;  // size (capacity) of stack
};
```

(1) Write appropriate C++ code for following copy constructor.

```
TIntStack::TIntStack(const TIntStack& that)
{

}
```

5. (32points) The class for "complex number" (복소수) can be often defined using operator overloading. Write your implementation of member functions (1) operator+ and (2) operator<<.
In (1) and (2), you may use standard library in your code. Your code should be grammatically and logically correct.

```
class Complex {
private:
   double re, im;
public :
   Complex( double r, double i ) : re(r), im(i) {}
   friend Complex operator+(Complex&, Complex&);
   friend Complex operator-(Complex&, Complex&);
   friend Complex operator*(Complex&, Complex&);
   friend ostream& operator<<(ostream &o, const Complex &x);
         // prints x in "a+bi" format. for example, if re=3.2 and im=2.3, then this fucntion should print 3.2+2.3i
         // if re=3.2 and im=-2.3, then this function should print 3.2-2.3i
         // if re=3.2 and im=0, then this function should print 3.2
};
```

(1) Write C++ code for the **operator+** function

```
Complex operator+(Complex& x, Complex& y)
{

}
```

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

```
ostream& operator<<(ostream &o , const Complex& x)
{

}
```