THE HONG KONG UNIVERSITY OF SCIENCE & TECHNOLOGY
Department of Computer Science & Engineering

COMP 2012: Object-Oriented Programming and Data Structures
Spring 2013

Midterm Examination

Date: Saturday, 13 April 2013
Time: 10:30pm – 12:00pm
Venue: LTD (L1), LTC (L2) and LTJ (L3)

- **This is a closed-book examination. However, you are allowed to bring with you a piece of A4-sized paper with notes written, drawn or typed on both sides for use in the examination.**
- **Your answers will be graded according to correctness, efficiency, precision, and clarity.**
- **During the examination, you must put aside your calculators, mobile phones, PDAs and all other electronic devices. All mobile phones must be turned off.**
- **You may use the reverse side of the pages for your rough work or continuation of work. This booklet has 12 single-sided pages. Please check that all pages are properly printed before you start working.**

Student name: ____________________ **English nickname (if any):** __________

Student ID: ____________________ **Email:** __________ **Lecture:** __________

I have not violated the Academic Honor Code in this examination (signature): ____________________

Question	Your score	Maximum score	Question	Your score	Maximum score
1		10	6		10
2		10	7		10
3		8	8		10
4		12	9		10
5		10	10		10
Total					100

A Vector Class

(100 points total)

In mathematics classes, you may have learnt vector. In this problem, you are to develop a `Vector` class holding `double` type supporting I/O and operator overloading. A vector with $s \geq 1$ components is an ordered collection of s doubles given by

$$v = (v_0, v_1, \ldots, v_{s-1}).$$

s is also called the size or dimension of the vector. Note that we index the components from 0 onwards. We define the case $s = 0$ a null or empty vector.

You are to implement a `Vector` class with a dynamic array `double *data` which holds the components, and a field `int size` which is the dimension of the vector.

We show below the C++ header file `Vector.h`, which defines the class `Vector` as follows:

```
#include <iostream>
#include <sstream>
#include <fstream>
using namespace std;

class Vector
{
public:
    Vector(int s=0, double v=0); //Constructor
    Vector(const Vector& other_vector);
    ~Vector();

    const Vector& operator = (const Vector&); //assignment
    double& operator [] (int);
    const double& operator [] (int) const;

    Vector operator+(const Vector&) const; //overloading of add

    void set_size(int d){ (d<0)? exit(-1): size=d; } //set the size of a vector
    int get_size() const{ return size; } //get the size of a vector

    void read_vector(const char* fileName);  //read vector from a file
    void write_vector(const char* fileName) const; //write vector to a file

private:
    int size; //dimensions of the vector
    double *data; // pointer to vector components

    // utility/helper functions
    void display(ostream& out) const; //display a vector to out
    void readin(istream& in); //read a vector from in

};
```

Given the above `Vector.h`, you are to write `Vector.cpp` so as to separate implementation details from function usages for separate compilation. In your implementation, you may exit your program with code -1 upon any error. Answer the following questions.

1. *(10 points)*

   (a) *(8 points)* In `Vector.cpp` file, show your implementation of the constructor:

   ```
    Vector(int s=0, double v=0);
   ```

   which simply constructs a *s*-dimensional `Vector` object with value *v* for all components. The default is to create a null vector. As an example, the code

   ```
   Vector v(4, 3.5);
   ```

   creates a vector $v = (3.5, 3.5, 3.5, 3.5)$.

   (b) *(2 points)* Please give below the vector the constructor creates with a call

   ```
   Vector v(3);
   ```

2. *(10 points)*

(a) *(2 points)* What does the following statement do?

```
Vector(const Vector& other_vector);
```

(b) *(8 points)* Please show your implementation codes of the above in `Vector.cpp`:

3. *(8 points)*

   (a) *(3 points)* What is the name of the following statement? What is it used for?

   ```
   ~Vector();
   ```

   (b) *(5 points)* Show your implementation codes of the above statement in `Vector.cpp`:

4. *(12 points)* In `Vector.cpp`, show your implementation of the assignment function

```
 const Vector& operator = (const Vector&);
```

which assigns the values of a `Vector` object to the current object.

5. *(10 points)*

(a) *(8 points)* Show your codes to implement the member function
```
const double& operator [] (int) const;
```
which overloads the `[]` operator to support the following statements:

```
const Vector v(10);
double d = v[3]; // return the fourth component, i.e., v3
```

(b) *(2 points)* Why are the following two functions?
```
double& operator [] (int);
const double& operator [] (int)const;
```

6. *(10 points)* Let $v = (v_0, v_1, \ldots, v_{s-1})$ and $u = (u_0, u_1, \ldots, u_{s-1})$ be two vectors of the same dimension. Let $w = v + u$ be the sum of v and u. The components of w are then given by $w_i = v_i + u_i$, for $0 \le i \le s - 1$.

   Implement the member function

   ```
   Vector operator+(const Vector&) const;
   ```

   which supports the addition of two or more vectors.

7. *(10 points)* Suppose you would like to support the addition of a *scalar* to a vector such that each of the components of the vector is increased by the scalar value. The following shows an example:

```
double d = 3.5;
Vector v(2, 4.5); // v = (4.5, 4.5)
Vector u;

u = d + v; // results: u = (8.0, 8.0)
```

Show the implementation codes you need to add (in both `Vector.h` and `Vector.cpp` files) to achieve the above by overloading the operator `+`.

8. *(10 points)* The private utility function

   ```
   void readin(istream& in);
   ```

   takes an `istream` object `in`, and sets the current `Vector` object accordingly. The input format of `in` is

   ```
   s v1 v2 v3 ... v(s-1) ,
   ```

   i.e., the first number is the dimension of the vector, followed by the vector components.

   For example, an input of

   ```
   3 1 2 3
   ```

   means to set the dimension of the current `Vector` object to *3*, with components 1.0, 2.0, and 3.0.

   Show your implementation below in `Vector.cpp`.

9. *(10 points)* Suppose a file of name `fileName` has the same format according to what is given in Problem 8. Implement the public member function

```
 void read_vector(const char* fileName);
```

which modifies the current `Vector` object by opening and reading the file.

10. *(10 points)* Suppose you would like to support

```
Vector v;
cin >> v;
```

where the input format is according to what is stated in Problem 8.

Show the implementation codes you add of overloading the >> operator in order to achieve the above.