Mock-up Final Exam

DURATION: 120 minutes

Task 1. (***50 points***) Given the **List** header class as follows:

```
template <class T>
class List
{
    private:
        T **data;
        int n;
    public:
        List();
        List(int n) ;
        ~List() ;
        void insertAt(int x, T value) ;
        T getAt(int x);
        void sort();
        void unique();
        List operator + (const List &list);
        friend istream &operator >> (istream &is, List &list);
        friend ostream &operator << (ostream &os, List &list);
};
```

1. Complete your code based on the above header. (40 points)
    a. **List()** and **List(int n)** are constructor which create elements by using pointers.
    b. **~List()** is destructor which free up pointers in memory.
    c. **Overloading input >>** lets user can enter elements with command prompt. **Overloading output <<** will output the list in one line.
    d. **insertAt(int x, T value)** method to let user insert an element with **value** type T to position **x** in list.
    e. **getAt(int x)** method will return value at position **x** in list.
    f. **Overloading operator +** to concatenate a list with another list.
    g. **sort()** method will arrange the list in ascending order.
    h. **unique()** method will remove duplicated elements in list.
2. Create a main function to perform: (10 points)
    a. Create two integer list based on List class, then use **overloading input >>** to enter elements in list. Concatenate two lists above into one list by using operator +.
    b. Insert a new character to the end of list and print out the list by using **insertAt**() method. Then, sort the new list by using **sort()** method and remove duplicated values by using **unique()** method.

```
SAMPLE OUTPUT
Input number of element of list A: 3
Enter elements: 1 4 2

Input number of element of list B: 5
Enter elements: 3 1 6 2 5

List C = A + B: 1 4 2 3 1 6 2 5
List C after insert 6: 1 1 2 2 3 4 5 6
List C after sorting: 1 1 2 2 3 4 5 6
List C after unique: 1 2 3 4 5 6
```

Task 2. (***50 points***) Given the following **Employee** class:

```
class Employee
{
    private:
        string name;
        int age;
        double salary;
        double bonus;
        double advance;
        double total;
    public:
        Employee();
        Employee(string name, int age, double salary, double bonus, double advance);
        ~Employee();
        double getSalary();
        double getBonus();
        double getAdvance();
        double getTotal();
        void setSalary(double salary);
        void setBonus(double bonus);
        void setAdvance(double advance);
        void setTotal(double total);
        bool operator > (const Employee &employee);
        bool operator < (const Employee &employee);
        friend ostream &operator << (ostream &os, Employee &employee);
};
```

1. Complete **Employee** class follow the design, the **overloading output <<** function print the information of that employee in one line. **Overloading operator >, <** will compare the **family name** between two employees. (20 points)

2. In main function, create an **employee.txt** as following data and perform read and import employees into an employee list (student can use vector) (*10 points*):
3. Calculate the total for all employee by using these formulas: (10 points)

$$Bonus = 10\%\ of\ Revenue$$
$$Total = Salary + Bonus - Advance$$

4. Output the employee list into **output.txt** as ascending of **family name** as following template: (10 points)

employee.txt	Sample	output.txt
n	3	3
NameE1	Lin Jia-Hui	Chen Zhi-Da
AgeE1	28	47000
SalaryE1	40000	Lin Jia-Hui
RevenueE1	120000	20000
AdvanceE1	32000	Yang Zhe-Wei
NameE2	Chen Zhi-Da	51000
AgeE2	24	
SalaryE2	38000	
RevenueE2	90000	
AdvanceE2	0	
NameE3	Yang Zhe-Wei	
AgeE3	26	
SalaryE3	45000	
RevenueE3	210000	
AdvanceE3	15000	