O-O Programming & Data Structure Lab. 7

7.1 Standard Template Library (STL) vector and algorithm for class Time and class Date

(1) class Time

```
/* Time.h */
#ifndef TIME_H
#define TIME_H
#include <iostream>
using namespace std;
class Time
{
	friend ostream& operator<<(ostream&, const Time&);
public:
	Time() { hour = 0; min = 0; sec = 0; } // default constructor
	Time(int h, int m, int s);
	int elasedSec() const;
	Time getTime() const { return Time(hour, min, sec); }
	bool operator<(const Time&) const;
	bool operator<=(const Time&) const;
	bool operator>(const Time&) const;
	bool operator>=(const Time&) const;
	bool operator==(const Time&) const;
private:
	bool isValidTime(int, int, int);
	int hour;
	int min;
	int sec;
};
#endif

// implement member functions here !!
```

(2) class Date

```
/* Date.h */
#ifndef DATE_H
#define DATE_H
#include <iostream>
using namespace std;
#define WEEKDAY_AD01Jan01 MON // the weekday of AD Jan 1.
#define DAYS_PER_WEEK 7
class Date
{
	friend ostream& operator<<(ostream&, const Date&);
public:
	Date();    // default constructor
	Date(int y, int m, int d); // constructor
	~Date(); // destructor
	int getWeekDay();
	int getElapsedDaysFromAD010101() const; //    get elapsed days from AD 1. 1. 1.
	int getElapsedDaysFromAD010101(Date) const;
	bool operator<(const Date&) const;
	bool operator<=(const Date&) const;
	bool operator>(const Date&) const;
	bool operator>=(const Date&) const;
	bool operator==(const Date&) const;
private:
	bool isValidDate(int y, int m, int d);
	int year;
	int month;
	int day;
};
bool isLeapYear(int y); // check whether the given year y is leap year
int getYearDay(int year, int month, int day);
#endif
```

(3) class VectorHandler

```
/* VectorHandler*/
#ifndef VECTOR_HANDLER_H
#define VECTOR_HANDLER_H
#include <vector>
#include <algorithm>
template<typename T>
void printVector(vector<T>& v)
{
	string typeName = typeid(T).name();
	cout << "Vector size(" << v.size() << "), elements : \n";
	typename vector<T>::iterator p;
	for (p = v.begin(); p != v.end(); p++)
	{
		cout << *p << " ";
		if ((typeName == "class Date") || (typeName == "class Time"))
			continue;
		else
			cout << endl;
	}
	cout << endl;
}
#endif
```

(4) main() with STL vector and algorithm

```
/** main - T_Array<Student, K> */
#include <iostream>
#include <conio.h> # for _getch()
#include <fstream>
#include <string>
#include <vector>
#include <algorithm>
#include "VectorHandler.h"
using namespace std;
#define NUM_TIMES 10
#define NUM_DATES 10
#define NUM_STUDENTS 10
void main()
{
	Time times[NUM_TIMES] =
	{
		Time(3, 0, 30), Time(7, 30, 0), Time(2, 0, 50), Time(5, 30, 0), Time(1, 10, 0),
		Time(9, 20, 10), Time(1, 20, 15), Time(10, 0, 0), Time(11, 15, 10), Time(2, 0, 5)
	};

	vector<Time> v_times(times, times+ NUM_TIMES);
	cout << "Initial v_times : " << endl;
	printVector(v_times);
	sort(v_times.begin(), v_times.end());
	cout << "After sort() : " << endl;
	printVector(v_times);
	Date dates[10] =
	{
		Date(2003, 4, 5), Date(2002, 7, 15), Date(2001, 5, 1), Date(2001, 3, 10), Date(2000, 5, 21),
		Date(2000, 3, 1), Date(1999, 12, 25), Date(1998, 10, 9), Date(1997, 6, 10), Date(1996, 1, 1)
	};
	vector<Date> v_dates(dates, dates + NUM_DATES);
	cout << "Initial v_dates :" << endl;
	printVector(v_dates);
	sort(v_dates.begin(), v_dates.end());
	cout << "After sort() : " << endl;
	printVector(v_dates);
	cout << "Hit any key to continue .... ";
	_getch();
	cout << endl;
}
```

(5) 실행결과

```
Initial v_times :
Vector size(10), elements :
( 3: 0:30) ( 7:30: 0) ( 2: 0:50) ( 5:30: 0) ( 1:10: 0) ( 9:20:10) ( 1:20:15) (10: 0: 0) (11:15:10) ( 2: 0: 5)
After sort() :
Vector size(10), elements :
( 1:10: 0) ( 1:20:15) ( 2: 0: 5) ( 2: 0:50) ( 3: 0:30) ( 5:30: 0) ( 7:30: 0) ( 9:20:10) (10: 0: 0) (11:15:10)
Initial v_dates :
Vector size(10), elements :
(2003. 4. 5) (2002. 7.15) (2001. 5. 1) (2001. 3.10) (2000. 5.21) (2000. 3. 1) (1999.12.25) (1998.10. 9) (1997. 6.10) (1996. 1. 1)
After sort() :
Vector size(10), elements :
(1996. 1. 1) (1997. 6.10) (1998.10. 9) (1999.12.25) (2000. 3. 1) (2000. 5.21) (2001. 3.10) (2001. 5. 1) (2002. 7.15) (2003. 4. 5)
Hit any key to continue ....
```

7.2 STL vector and algorithm for class Student that inherits class Person

(1) class Person

```
/* Person.h */
#ifndef PERSON_H
#define PERSON_H
#include <iostream>
#include <string>
#include "Date.h"
#include "Time.h"
class Person
{
	friend ostream& operator<< (ostream& fout, const Person& p)
	{
		fout << " Person [name: " << p.name << "]";
		return fout;
	}
public:
	Person() { name = "nobody"; }
	Person(string n) { name = n; }
	void setName(string n) { name = n; }
	string getName() { return name; }
	void setDoB(Date dob) { dateOfBirth = dob; }
	const Date getDoB() const { return dateOfBirth; }
	void setArrivalTime(Time t) { arrivalTime = t; }
	const Time getArrivalTime() const { return arrivalTime; }
protected:
	string name;
	Date dateOfBirth; // date of birth
	Time arrivalTime;
};
#endif
```

(2) class Student

```
/* Student.h */
#ifndef STUDENT_H
#define STUDENT_H
#include "Person.h"
#include "Date.h"
#include "Time.h"
class Student : public Person
{
	friend ostream & operator<< (ostream &, Student &);
public:
	Student();   // default constructor
	Student(int s_id, string n, Date dob, Time avt, double gpa);
	void getKey(string keyName, void* pKey);
	bool operator<(const Student&) const;
	bool operator<=(const Student&) const;
	bool operator>(const Student&) const;
	bool operator>=(const Student&) const;
	bool operator==(const Student&) const;
```

```
private:
        int st_id;
        double gpa;
};
#endif
```

(3) main() with STL vector and algorithm for class Student

```
/** main – STL vector and algorithm for class Student */
#include <iostream>
#include <conio.h> # for _getch()
#include <fstream>
#include <string>
#include <vector>
#include <algorithm>
#include "VectorHandler.h"
using namespace std;
#define NUM_TIMES 10
#define NUM_DATES 10
#define NUM_STUDENTS 10
void main()
{
        Student students[NUM_STUDENTS] =
        {
                Student(5234, string("Kim,    G-M"), Date(2002, 7, 15), Time(3, 0, 30), 3.57),
                Student(1999, string("Yoon, S-M"), Date(1999, 12, 25), Time(7, 30, 0),    4.37),
                Student(4141, string("Byun, S-S"), Date(2001, 3, 10), Time(2, 0, 50), 2.72),
                Student(2167, string("Lee,    K-M"), Date(1998, 10, 9), Time(5, 30, 0), 3.35),
                Student(3890, string("Hong, G-M"), Date(2000, 3, 1), Time(1, 10, 0),    3.89),
                Student(6543, string("Jang, S-M"), Date(2000, 5, 21), Time(9, 20, 10),    4.42),
                Student(7080, string("Park, S-T"), Date(2001, 5, 1), Time(1, 20, 15),    4.12),
                Student(9564, string("Choi, Y-H"), Date(1997, 6, 10), Time(10, 0, 0),    3.85),
                Student(1000, string("Shin, D-J"), Date(2003, 4, 5), Time(11, 15, 10),    3.21),
                Student(8812, string("Ahn,    S-B"), Date(1997, 1, 1), Time(2, 0, 5), 4.45),
        };

        vector<Student> v_students(students, students + NUM_DATES);
        cout << "Initial v_students :" << endl;
        printVector(v_students);
        sort(v_students.begin(), v_students.end());
        cout << "₩nstudents after sorting by st_id :" << endl;
        printVector(v_students);
        cout << "Hit any key to continue .... ";
        _getch();
        cout << endl;
}
```

(4) 실행결과

```
Initial v_students :
Vector size(10), elements :
Student [ st_id:  5234, name: Kim,  G-M, gpa: 3.57 , date_of_birth: (2002. 7.15), arrival: ( 3: 0:30)]
Student [ st_id:  1999, name: Yoon, S-M, gpa: 4.37 , date_of_birth: (1999.12.25), arrival: ( 7:30: 0)]
Student [ st_id:  4141, name: Byun, S-S, gpa: 2.72 , date_of_birth: (2001. 3.10), arrival: ( 2: 0:50)]
Student [ st_id:  2167, name: Lee,  K-M, gpa: 3.35 , date_of_birth: (1998.10. 9), arrival: ( 5:30: 0)]
Student [ st_id:  3890, name: Hong, G-M, gpa: 3.89 , date_of_birth: (2000. 3. 1), arrival: ( 1:10: 0)]
Student [ st_id:  6543, name: Jang, S-M, gpa: 4.42 , date_of_birth: (2000. 5.21), arrival: ( 9:20:10)]
Student [ st_id:  7080, name: Park, S-T, gpa: 4.12 , date_of_birth: (2001. 5. 1), arrival: ( 1:20:15)]
Student [ st_id:  9564, name: Choi, Y-H, gpa: 3.85 , date_of_birth: (1997. 6.10), arrival: (10: 0: 0)]
Student [ st_id:  1000, name: Shin, D-J, gpa: 3.21 , date_of_birth: (2003. 4. 5), arrival: (11:15:10)]
Student [ st_id:  8812, name: Ahn,  S-B, gpa: 4.45 , date_of_birth: (1997. 1. 1), arrival: ( 2: 0: 5)]


students after sorting by st_id :
Vector size(10), elements :
Student [ st_id:  1000, name: Shin, D-J, gpa: 3.21 , date_of_birth: (2003. 4. 5), arrival: (11:15:10)]
Student [ st_id:  1999, name: Yoon, S-M, gpa: 4.37 , date_of_birth: (1999.12.25), arrival: ( 7:30: 0)]
Student [ st_id:  2167, name: Lee,  K-M, gpa: 3.35 , date_of_birth: (1998.10. 9), arrival: ( 5:30: 0)]
Student [ st_id:  3890, name: Hong, G-M, gpa: 3.89 , date_of_birth: (2000. 3. 1), arrival: ( 1:10: 0)]
Student [ st_id:  4141, name: Byun, S-S, gpa: 2.72 , date_of_birth: (2001. 3.10), arrival: ( 2: 0:50)]
Student [ st_id:  5234, name: Kim,  G-M, gpa: 3.57 , date_of_birth: (2002. 7.15), arrival: ( 3: 0:30)]
Student [ st_id:  6543, name: Jang, S-M, gpa: 4.42 , date_of_birth: (2000. 5.21), arrival: ( 9:20:10)]
Student [ st_id:  7080, name: Park, S-T, gpa: 4.12 , date_of_birth: (2001. 5. 1), arrival: ( 1:20:15)]
Student [ st_id:  8812, name: Ahn,  S-B, gpa: 4.45 , date_of_birth: (1997. 1. 1), arrival: ( 2: 0: 5)]
Student [ st_id:  9564, name: Choi, Y-H, gpa: 3.85 , date_of_birth: (1997. 6.10), arrival: (10: 0: 0)]

Hit any key to continue ....
```

7.3 Generic Template Class T_Array for class Student

(1) class T_Array

```
/* Template class T_Array.h */
#ifndef T_Array_H
#define T_Array_H
#include <iostream>
#include <iomanip>
#include "Date.h"
#include "Time.h"
using namespace std;
enum SortingOrder { INCREASING, DECREASING };

template<typename T, typename K>
class T_Array
{
public:
    T_Array(int n, string nm); // constructor
    ~T_Array(); // destructor
    int size() { return num_elements; }
    bool empty() { return num_elements == 0; }
    string getName() { return name; }
    void insert(int i, T element);
    int sequential_search(K search_key); // search and return the index; -1 if not found
    int binary_search(K search_key); // search and return the index; -1 if not found
    void selection_sort(string keyName, SortingOrder sortOrder);
    void print(int elements_per_line);
    bool isValidIndex(int i);
    T& operator[](int index) { return t_array[index]; }
private:
    T *t_array;
    int num_elements;
    int capacity;
    string name;
};

template<typename T, typename K>
T_Array<T, K>::T_Array(int new_capacity, string nm) // constructor
{
    // . . . .
}

template<typename T, typename K>
T_Array<T, K>::~T_Array() // destructor
{
    // . . . .
}


template<typename T, typename K>
bool T_Array<T, K>::isValidIndex(int index)
{
    // . . . .
}

template<typename T, typename K>
void T_Array<T, K>::insert(int i, T new_element)
{
    // . . . .
}

template<typename T, typename K>
int T_Array<T, K>::sequential_search(K search_key)
{
    // . . . .
}
```

```
template<typename T, typename K>
int T_Array<T, K>::binary_search(K search_key)
{
      // . . . .
}

template<typename T, typename K>
void T_Array<T, K>::selection_sort(string keyName, SortingOrder sortOrder)
{
    // . . . .
}

template<typename T, typename K>
void T_Array<T, K>::print(int elements_per_line)
{
    int count = 0;
    while (count < num_elements)
    {
            for (int i = 0; i < elements_per_line; i++)
            {
                    cout << t_array[count] << "     ";
                    count++;
                    if (count % elements_per_line == 0)
                            cout << endl;
            }
    }
    cout << endl;
}
#endif
```

(2) Template Class T_Array의 generic algorithm 사용을 위한 class Student의 기능 추가

```
void Student::getKey(string keyName, void* pKey)
{
      if (keyName == "ST_ID")
            *(int *)pKey = this->st_id;
      else if (keyName == "ST_NAME")
            *(string *)pKey = this->name;
      else if (keyName == "GPA")
            *(double *)pKey = this->gpa;
      else if (keyName == "ARRIVAL_TIME")
            *(Time *)pKey = this->arrivalTime;
      else if (keyName == "BIRTH_DATE")
            *(Date *)pKey = this->dateOfBirth;
      else
            pKey = NULL;
}
```

(3) main()

```
/** main - T_Array<Student, K>    (1) */
#include <iostream>
#include <conio.h> # for _getch()
#include <fstream>
#include <string>
#include <vector>
#include <algorithm>
#include "T_Array.h"
#include "Student.h"
#include "VectorHandler.h"
using namespace std;
#define STEP_1
#define STEP_2
#define STEP_3
#define STEP_4
#define NUM_TIMES 10
```

```
#define NUM_DATES 10
#define NUM_STUDENTS 10
void main()
{
	Student students[NUM_STUDENTS] =
	{
		Student(5234, string("Kim,    G-M"), Date(2002, 7, 15), Time(3, 0, 30), 3.57),
		Student(1999, string("Yoon, S-M"), Date(1999, 12, 25), Time(7, 30, 0),    4.37),
		Student(4141, string("Byun, S-S"), Date(2001, 3, 10), Time(2, 0, 50), 2.72),
		Student(2167, string("Lee,    K-M"), Date(1998, 10, 9), Time(5, 30, 0), 3.35),
		Student(3890, string("Hong, G-M"), Date(2000, 3, 1), Time(1, 10, 0),    3.89),
		Student(6543, string("Jang, S-M"), Date(2000, 5, 21), Time(9, 20, 10),    4.42),
		Student(7080, string("Park, S-T"), Date(2001, 5, 1), Time(1, 20, 15),    4.12),
		Student(9564, string("Choi, Y-H"), Date(1997, 6, 10), Time(10, 0, 0),    3.85),
		Student(1000, string("Shin, D-J"), Date(2003, 4, 5), Time(11, 15, 10),    3.21),
		Student(8812, string("Ahn,    S-B"), Date(1997, 1, 1), Time(2, 0, 5), 4.45),
	};

#ifdef STEP_3
	/* Step 3 - Testing template array class T_Array<T, K> */
	Student* pSt;
	T_Array<Student, int> stArray_keyID(NUM_STUDENTS, "T_Array<Student, keyST_ID>");
	for (int i = 0; i < NUM_STUDENTS; i++)
	{
		stArray_keyID.insert(i, students[i]);
	}
	cout << "T_Array<Student_keyID> at initialization : " << endl;
	stArray_keyID.print(1);
	stArray_keyID.selection_sort(string("ST_ID"), INCREASING);
	cout << "\nT_Array<Student_keyID> after sorting (increasing order) by ST_ID : " << endl;
	stArray_keyID.print(1);
	T_Array<Student, double> stArray_keyGPA(NUM_STUDENTS, "T_Array<Student, keyGPA>");
	for (int i = 0; i < NUM_STUDENTS; i++)
	{
		stArray_keyGPA.insert(i, students[i]);
	}
	stArray_keyGPA.selection_sort(string("GPA"), DECREASING);
	cout << "\nT_Array<Student, keyGPA> after sorting (decreasing order) by GPA : " << endl;
	stArray_keyGPA.print(1);
	cout << "Hit any key to continue .... ";
	_getch();
	cout << endl;
#endif

#ifdef STEP_4
	T_Array<Student, string> stArray_keyName(NUM_STUDENTS, "T_Array<Student,
		keyName>");
	for (int i = 0; i < NUM_STUDENTS; i++)
	{
		stArray_keyName.insert(i, students[i]);
	}
	stArray_keyName.selection_sort(string("ST_NAME"), INCREASING);
	cout << "\nT_Array<Student_keyName> after sorting (increasing order) by name : " << endl;
	stArray_keyName.print(1);
	T_Array<Student, Date> stArray_keyDoB(NUM_STUDENTS,
		"Array of Students, date of birth as key");
	for (int i = 0; i < NUM_STUDENTS; i++)
	{
		stArray_keyDoB.insert(i, students[i]);
	}
	stArray_keyDoB.selection_sort(string("BIRTH_DATE"), INCREASING);
	cout << "\nstArray_keyDoB after sorting (increasing order) by date of birth : " << endl;
	stArray_keyDoB.print(1);
	T_Array<Student, Time> stArray_keyTime(NUM_STUDENTS,
		"Array of Students, arrival time as key");
	for (int i = 0; i < NUM_STUDENTS; i++)
```

```
        {
                stArray_keyTime.insert(i, students[i]);
        }
        stArray_keyTime.selection_sort(string("ARRIVAL_TIME"), INCREASING);
        cout << "\nstArray_keyArrTm after sorting (increasing order) by arrival time : " << endl;
        stArray_keyTime.print(1);
#endif
}
```

(4) 실행 결과 (step_3)

```
T_Array<Student_keyID> at initialization :
Student [ st_id:  5234, name: Kim,  G-M, gpa: 3.57 , date_of_birth: (2002. 7.15), arrival: ( 3: 0:30)]
Student [ st_id:  1999, name: Yoon, S-M, gpa: 4.37 , date_of_birth: (1999.12.25), arrival: ( 7:30: 0)]
Student [ st_id:  4141, name: Byun, S-S, gpa: 2.72 , date_of_birth: (2001. 3.10), arrival: ( 2: 0:50)]
Student [ st_id:  2167, name: Lee,  K-M, gpa: 3.35 , date_of_birth: (1998.10. 9), arrival: ( 5:30: 0)]
Student [ st_id:  3890, name: Hong, G-M, gpa: 3.89 , date_of_birth: (2000. 3. 1), arrival: ( 1:10: 0)]
Student [ st_id:  6543, name: Jang, S-M, gpa: 4.42 , date_of_birth: (2000. 5.21), arrival: ( 9:20:10)]
Student [ st_id:  7080, name: Park, S-T, gpa: 4.12 , date_of_birth: (2001. 5. 1), arrival: ( 1:20:15)]
Student [ st_id:  9564, name: Choi, Y-H, gpa: 3.85 , date_of_birth: (1997. 6.10), arrival: (10: 0: 0)]
Student [ st_id:  1000, name: Shin, D-J, gpa: 3.21 , date_of_birth: (2003. 4. 5), arrival: (11:15:10)]
Student [ st_id:  8812, name: Ahn,  S-B, gpa: 4.45 , date_of_birth: (1997. 1. 1), arrival: ( 2: 0: 5)]


T_Array<Student_keyID> after sorting (increasing order) by ST_ID :
Student [ st_id:  1000, name: Shin, D-J, gpa: 3.21 , date_of_birth: (2003. 4. 5), arrival: (11:15:10)]
Student [ st_id:  1999, name: Yoon, S-M, gpa: 4.37 , date_of_birth: (1999.12.25), arrival: ( 7:30: 0)]
Student [ st_id:  2167, name: Lee,  K-M, gpa: 3.35 , date_of_birth: (1998.10. 9), arrival: ( 5:30: 0)]
Student [ st_id:  3890, name: Hong, G-M, gpa: 3.89 , date_of_birth: (2000. 3. 1), arrival: ( 1:10: 0)]
Student [ st_id:  4141, name: Byun, S-S, gpa: 2.72 , date_of_birth: (2001. 3.10), arrival: ( 2: 0:50)]
Student [ st_id:  5234, name: Kim,  G-M, gpa: 3.57 , date_of_birth: (2002. 7.15), arrival: ( 3: 0:30)]
Student [ st_id:  6543, name: Jang, S-M, gpa: 4.42 , date_of_birth: (2000. 5.21), arrival: ( 9:20:10)]
Student [ st_id:  7080, name: Park, S-T, gpa: 4.12 , date_of_birth: (2001. 5. 1), arrival: ( 1:20:15)]
Student [ st_id:  8812, name: Ahn,  S-B, gpa: 4.45 , date_of_birth: (1997. 1. 1), arrival: ( 2: 0: 5)]
Student [ st_id:  9564, name: Choi, Y-H, gpa: 3.85 , date_of_birth: (1997. 6.10), arrival: (10: 0: 0)]


T_Array<Student, keyGPA> after sorting (decreasing order) by GPA :
Student [ st_id:  8812, name: Ahn,  S-B, gpa: 4.45 , date_of_birth: (1997. 1. 1), arrival: ( 2: 0: 5)]
Student [ st_id:  6543, name: Jang, S-M, gpa: 4.42 , date_of_birth: (2000. 5.21), arrival: ( 9:20:10)]
Student [ st_id:  1999, name: Yoon, S-M, gpa: 4.37 , date_of_birth: (1999.12.25), arrival: ( 7:30: 0)]
Student [ st_id:  7080, name: Park, S-T, gpa: 4.12 , date_of_birth: (2001. 5. 1), arrival: ( 1:20:15)]
Student [ st_id:  3890, name: Hong, G-M, gpa: 3.89 , date_of_birth: (2000. 3. 1), arrival: ( 1:10: 0)]
Student [ st_id:  9564, name: Choi, Y-H, gpa: 3.85 , date_of_birth: (1997. 6.10), arrival: (10: 0: 0)]
Student [ st_id:  5234, name: Kim,  G-M, gpa: 3.57 , date_of_birth: (2002. 7.15), arrival: ( 3: 0:30)]
Student [ st_id:  2167, name: Lee,  K-M, gpa: 3.35 , date_of_birth: (1998.10. 9), arrival: ( 5:30: 0)]
Student [ st_id:  1000, name: Shin, D-J, gpa: 3.21 , date_of_birth: (2003. 4. 5), arrival: (11:15:10)]
Student [ st_id:  4141, name: Byun, S-S, gpa: 2.72 , date_of_birth: (2001. 3.10), arrival: ( 2: 0:50)]

Hit any key to continue ....
```

(5) 실행 결과 (step_4)

```
T_Array<Student_keyName> after sorting (increasing order) by name :
Student [ st_id:  8812, name: Ahn,  S-B, gpa: 4.45 , date_of_birth: (1997. 1. 1), arrival: ( 2: 0: 5)]
Student [ st_id:  4141, name: Byun, S-S, gpa: 2.72 , date_of_birth: (2001. 3.10), arrival: ( 2: 0:50)]
Student [ st_id:  9564, name: Choi, Y-H, gpa: 3.85 , date_of_birth: (1997. 6.10), arrival: (10: 0: 0)]
Student [ st_id:  3890, name: Hong, G-M, gpa: 3.89 , date_of_birth: (2000. 3. 1), arrival: ( 1:10: 0)]
Student [ st_id:  6543, name: Jang, S-M, gpa: 4.42 , date_of_birth: (2000. 5.21), arrival: ( 9:20:10)]
Student [ st_id:  5234, name: Kim,  G-M, gpa: 3.57 , date_of_birth: (2002. 7.15), arrival: ( 3: 0:30)]
Student [ st_id:  2167, name: Lee,  K-M, gpa: 3.35 , date_of_birth: (1998.10. 9), arrival: ( 5:30: 0)]
Student [ st_id:  7080, name: Park, S-T, gpa: 4.12 , date_of_birth: (2001. 5. 1), arrival: ( 1:20:15)]
Student [ st_id:  1000, name: Shin, D-J, gpa: 3.21 , date_of_birth: (2003. 4. 5), arrival: (11:15:10)]
Student [ st_id:  1999, name: Yoon, S-M, gpa: 4.37 , date_of_birth: (1999.12.25), arrival: ( 7:30: 0)]


stArray_keyDoB after sorting (increasing order) by date of birth :
Student [ st_id:  8812, name: Ahn,  S-B, gpa: 4.45 , date_of_birth: (1997. 1. 1), arrival: ( 2: 0: 5)]
Student [ st_id:  9564, name: Choi, Y-H, gpa: 3.85 , date_of_birth: (1997. 6.10), arrival: (10: 0: 0)]
Student [ st_id:  2167, name: Lee,  K-M, gpa: 3.35 , date_of_birth: (1998.10. 9), arrival: ( 5:30: 0)]
Student [ st_id:  1999, name: Yoon, S-M, gpa: 4.37 , date_of_birth: (1999.12.25), arrival: ( 7:30: 0)]
Student [ st_id:  3890, name: Hong, G-M, gpa: 3.89 , date_of_birth: (2000. 3. 1), arrival: ( 1:10: 0)]
Student [ st_id:  6543, name: Jang, S-M, gpa: 4.42 , date_of_birth: (2000. 5.21), arrival: ( 9:20:10)]
Student [ st_id:  4141, name: Byun, S-S, gpa: 2.72 , date_of_birth: (2001. 3.10), arrival: ( 2: 0:50)]
Student [ st_id:  7080, name: Park, S-T, gpa: 4.12 , date_of_birth: (2001. 5. 1), arrival: ( 1:20:15)]
Student [ st_id:  5234, name: Kim,  G-M, gpa: 3.57 , date_of_birth: (2002. 7.15), arrival: ( 3: 0:30)]
Student [ st_id:  1000, name: Shin, D-J, gpa: 3.21 , date_of_birth: (2003. 4. 5), arrival: (11:15:10)]


stArray_keyArrTm after sorting (increasing order) by arrival time :
Student [ st_id:  3890, name: Hong, G-M, gpa: 3.89 , date_of_birth: (2000. 3. 1), arrival: ( 1:10: 0)]
Student [ st_id:  7080, name: Park, S-T, gpa: 4.12 , date_of_birth: (2001. 5. 1), arrival: ( 1:20:15)]
Student [ st_id:  8812, name: Ahn,  S-B, gpa: 4.45 , date_of_birth: (1997. 1. 1), arrival: ( 2: 0: 5)]
Student [ st_id:  4141, name: Byun, S-S, gpa: 2.72 , date_of_birth: (2001. 3.10), arrival: ( 2: 0:50)]
Student [ st_id:  5234, name: Kim,  G-M, gpa: 3.57 , date_of_birth: (2002. 7.15), arrival: ( 3: 0:30)]
Student [ st_id:  2167, name: Lee,  K-M, gpa: 3.35 , date_of_birth: (1998.10. 9), arrival: ( 5:30: 0)]
Student [ st_id:  1999, name: Yoon, S-M, gpa: 4.37 , date_of_birth: (1999.12.25), arrival: ( 7:30: 0)]
Student [ st_id:  6543, name: Jang, S-M, gpa: 4.42 , date_of_birth: (2000. 5.21), arrival: ( 9:20:10)]
Student [ st_id:  9564, name: Choi, Y-H, gpa: 3.85 , date_of_birth: (1997. 6.10), arrival: (10: 0: 0)]
Student [ st_id:  1000, name: Shin, D-J, gpa: 3.21 , date_of_birth: (2003. 4. 5), arrival: (11:15:10)]
```

7.5 Oral Test

(1) 클래스 템플릿이 무엇이며, 클래스 템플릿을 사용하는 장점과 단점에 대하여 예를 들어 설명하라.

(2) STL (Standard Template Library)는 무엇이며, 어떤 기능 (자료구조 또는 알고리즘)이 제공되는가에 대하여 표를 만들어 설명하라. (Keypoints: container, algorithm, iterator)

(3) STL vector, STL list, STL deque 이 제공하는 주요 멤버함수 들에 대하여 설명하라.

(4) 알고리즘 추상화 (algorithm abstraction)는 무엇이며, 알고리즘 추상화를 어떻게 구현하는가를 대하여 퀵 정렬을 예로 들어 설명하라. 이 일반화된 알고리즘이 다수의 속성을 가지는 클래스 배열에 사용될 수 있게 하고, 알고리즘에 사용되는 기준 키의 이름 (keyName)이 알고리즘 호출에 전달되는 구조를 사용할 것.