Programming I and II Using C++ for Beginning IS Students

Mehdi Raoufi

and

John Maniotes

Information Systems and Computer Programming Department
Purdue University Calumet
Hammond, IN 46323 USA

Abstract

This paper describes the topics and objectives for a two-semester sequence, Programming I and II using the C++ language, for beginning students majoring in 2-year and 4-year IS programs. The paper explores the recent advances in the field of programming and how to incorporate these into the undergraduate IS curriculum. Object-oriented programming, object-oriented analysis and design, generic programming using STL, the necessary topics in data structures, and algorithms and their complexity required to understand STL are also explored.

Keywords: Algorithms, structured programming, Abstract Data Type (ADT), Object-Oriented Programming (OOP), container classes, generic functions, Standard Template Library (STL).

1. THE PROGRAMMING I COURSE

Purpose
Programming I is a one-semester course aimed at freshman level, IS students. This course teaches procedural programming, as well as an introduction to object-oriented programming, using the C++ language.

The authors will share some of their successful teaching techniques and lab exercises used to make the course challenging and interesting. Currently, this course is titled CIS 166 Introduction to Programming I Using C++ (2 hours lecture, 2 hours lab, 3 credits).

No previous experience with computer programming is assumed for Programming I. However, two semesters of high school algebra are required. Emphasis in Programming I is on procedural programming and top-down design. Topics include:

- Identifiers
- Data types
- Arithmetic and logical operations
- If, if/else, while, do/while, for, and switch statements
- Functions
- Arrays (single and multi-dimensional)
- Pointers
- Strings
- Struct and introduction to classes

Extensive programming exercises in C++ are assigned in a closed lab format. Currently, all Programming is performed on Intel x86 based PCs, such as the Pentium, using Microsoft Visual C++. In the past, we have also utilized Compaq/DEC VAX and ALPHA VMS systems using C++.

Objectives of Programming I
Upon completion of the course students will learn:

- Structured programming and its history, the top/down approach, and modular programming.
- Data types in C++, such as fixed-point and floating-point representations (single, double, and quad precision).

- Foundation in C++ for further studies in computer programming; control structures, functions, arrays, pointers, and structures.
- Searching and sorting algorithms.
- Recursive functions.
- How to develop structured modular programs which are error free.
- How to develop quality programs based on good software engineering practices.

The Lab Exercises for Programming I
Each semester we change the homework problems and lab exercises in order to keep to a minimum the number of "files" that students accumulate. However, the lab assignments for the closed lab primarily are designed for students to learn the C++ concepts and master various programming techniques. Several programs are assigned each week, and students are assisted in a closed lab format to complete them. Some of the homework problems and lab exercises that we have assigned have dealt with the following:

- Application of integer division and modulus
- Sequence of if statements and nested if statements
- Finding the largest or smallest value from a group of numbers.
- Calculating the factorials of non-negative integer numbers
- Application of De Morgan's Laws
- Nested loops
- Rounding and truncating of real numbers using floor and ceiling functions.
- Calculating the Greatest Common Divisor of two integer numbers (both iterative and recursive)
- Random function
- Calculating fibonacci sequence (both iterative and recursive)
- Tower of Hanoi and its efficiency
- The Quicksort algorithm to sort files
- Simulating the toss of two dice
- Pointer applications
- Application of functions (strcmp(), strcpy(), strtok(), etc.) to manipulate strings
- Application of classes; complex number as a class

Other problems are assigned as homework. These problems involve two to four assignments each semester, and are designed to improve students' skills in writing relatively large programs by combining what they have learned through out the course. An example of this type of problem is the Airline Reservation problem found in the Appendix of this paper. This problem is not only designed to make students apply the concepts of top-down design and modular programming, but also encourages students to master C++ functions and passing parameters.

Books and URLs for Programming I
The required textbook for Programming I is (Deitel 1998). However, we have also placed on reserve in the library the following recommended books that studends can use (Savitch 1999 and Dale 2000).

The internet is an excellent source of information on the latest news and developments dealing with the C++ language. During the semester, we encourage the students to use the following interesting URLs:

www.cuj.com/ (C++ Users Journal)
www.ddj.com/ (Dr. Dobbs Journal)
www.aul.fiu.edu/tech/visualc.html
(Visual C++ Resourses)
www.progsource.com/index.html
(Programmer's Source)

2. THE PROGRAMMING II COURSE

Purpose
Programming II is a one-semester course aimed at the second semester freshman level, IS students. This course teaches object-oriented programming using the C++ language. This course is also in a lecture-lab environment.

Currently, this course is titled CIS 266 Introduction to Programming II Using C++ (2 hours lecture, 2 hours lab, 3 credits). The prerequisite for this course is CIS 166 Introduction to Programming I. Topics include:

- Definition of classes
- Data abstraction
- Friend member functions
- The *this* pointer
- Static class member
- Operator overloading
- Inheritance
- Polymorphism, virtual functions, and abstract classes
- Reusability issues
- Template
- Exception handling
- Elementary data structures
- Introduction to the standard template library

Extensive programming exercises in C++ are assigned. All programming is performed using MS Visual C++ on Pentium based PCs.

Objectives of Programming II
Upon completion of the course students will learn:

- Abstract Data Types (ADTs)
- Classes, private data members, and public functions, encapsulation, and information hiding

- Object as an instance of a class
- Operator overloading, inheritance, and polymorphism
- Object Oriented Programming (OOP) paradigm
- Reuse of the existing implementation through inheritance
- Reuse of the existing public interface through virtual functions and dynamic binding
- Object-Oriented Analysis and Design (OOA&D)
- Elementary data structures, algorithms, and their complexity, necessary to understand the Standard Template Library (STL)
- Latest trends in programming; such as generic classes, generic algorithms, and STL

The Lab Exercises for Programming II
Some of the lab exercises that we have assigned in Programming II have dealt with the following:

- Class Point1, which represents a point in Cartesian coordinate system
- Class Point2, which represents a point in three-dimensional space
- A SalesPerson class which represents an array of weekly sales
- A class, which represents an integer set
- A class, which represent a saving account
- Inheritance: Employee class
- Inheritance: The Point, Circle, and Cylinder problem
- Composition: The Point, Circle, and Cylinder problem
- The class *list*: Array implementation
- The class *list*: Dynamic array implementation
- The class *list*: Linked structure implementation
- The class *list*: Template implementation
- String class as a first class object
- Application of vector, list, deque, stack, queue, set, and map from STL
- The Airline Reservation problem using object-oriented programming

Books and URLs for Programming II
The required textbook for Programming II is (Deitel 1998). We have also placed on reserve in the library the following recommended books that students can consult (Lafore 1999; Lippman 1998; Breymann 1998; Austern 1999; Musser 1996; and Main 1997).

The URLs for this course are the same as for Programming I. However, we have added other URLs so students may also see the latest C++ software related products from the following vendors:

www.microsoft.com
www.sun.con
www.sgi.com/Technology/STL
http://info.desy.de/gna/html/cc/index.html
(Introduction to OOP/C++ tutorial)
ftp.cs.rpi.edu/pub/stl
(Hewlett Packard Reference Implementation of STL)

3. RETENTION RATES OF IS MAJORS

The department has made a concentrated effort to retain IS majors from the freshman to the sophomore year in all courses, especially in CIS 166 and CIS 266. The department has provided for closed labs, instead of open labs, for these two courses.

Whenever possible, the department has assigned a full-time instructor, instead of a part-time instructor or a graduate student. In some instances, the department has also assigned a student assistant to help the instructor in the laboratory and to assist in grading the lab or homework exercises.

Currently, for these two courses, the lecture size is set to a maximum of 36 students and the labs are set to 18 students.

We have found that the retention rates generally are greater with the older, more mature students attending our **evening** classes than the typical young students who attend the **day** classes.

For the 1999 - 2000 academic fall, the retention rates were as follows:

1999 Fall Semester	**Number of Students**			
	Enrolled	Passing	Failing	%Retained
CIS 166 Programming I				
Day Division	20	14	1	15/20 = 75%
Evening Division	33	28	1	29/33 = 89%
CIS 266 Programming II				
Evening Division	13	10	0	10/13 = 77%

2000 Spring Semester

CIS 166 Programming I				
Distance Learning	9	5	1	6/9 = 67%
Evening Division	15	9	2	11/15 = 73%
CIS 266 Programming II				
Evening Division	20	12	0	12/20 = 60%

2000 Summer Semester **Number of Students**

	Enrolled	Passing	Failing	%Retained
CIS 266 Programming II				
Evening Division	12	10	1	11/12 = 92%

Most of these two programming courses have been taught using the traditional synchronous classroom instructional model. We have also experimented with the asynchronous distance learning (DL) model using small classes of 12 students to a class. The DL retention rates have been lower, and more work is required to increase these retention rates.

We feel that the department has also enriched the quality of student lab experiences by providing consistent maintenance and timely upgrades for the computer hardware and software.

4. FUTURE ENHANCEMENTS

In review of the many technical questions raised by the students in these two courses concerning the syntax and the semantics of the 1998 C++ standard, we have placed on reserve in the library for the 2000 Fall semester, a copy of the complete ISO/ANSI C++ standard. Although this document contains almost 800 pages and costs about $300, we feel this is a wise investment that will be of benefit to both the students and the instructors for Programming I and II.

The department has an Industrial Advisory Committee made up of about 10 representatives from local businesses and industry that have hired our IS graduates over the years. This committee meets every semester to review the IS curriculum, courses, options, laboratory needs, etc. At the next meeting, we will ask the committee for their feedback on these two programming courses, as well as other C++ topics that may be included, so that we may fine tune them accordingly.

5. CONCLUSIONS

These two courses are designed to provide students with a firm foundation in programming and the C++ language. Students successfully completing these two courses learn in depth both procedural programming and object-oriented programming. In the first course, students are introduced early to the top-down design approach, before they learn about looping and decision making, and in the second course, students are introduced early to the object-oriented programming paradigm, OO analysis and design, inheritance, and polymorphism.

The strength of the two courses lies in the homework problems and lab exercises assigned. With the variety of problems and exercises, students "learn" good programming habits and C++ by "doing it" in a lecture-lab environment.

6. REFERENCES

Austern, M. H., 1999, Generic Programming and the STL, Addison Wesley, Reading, MA.

Breymann, U., 1998, Designing Components with the C++ STL, Addison Wesley, Reading, MA.

Dale, N., C. Weems, and M. Headington,, 2000, Programming and Problem Solving with C++, Jonesand Bartlett, Boston, MA.

Deitel, H. M. and P. J. Deitel, 1998, C++ How to Program, Prentice Hall, Upper Saddle River, NJ.

Lafore, R., 1999, Object-Oriented Programming in C++, SAMS, Indianapolis, IN.

Lippman, S. B. and J. Lajoie, 1998, C++ Primer, Addison Wesley, Reading, MA.

Main, M. and W. Savitch,, 1997, Data Structures and other Objects using C++, Addison Wesley, MA.

Musser, D. R. and A. Saini, 1996, STL Tutorial and Reference Guide C++ Programming with the Standard Template, Addison Wesley, Reading, MA.

Savitch , W., 1999, Problem Solving with C++ The Object of Programming, Addison Wesley, Reading, MA.

7. APPENDIX

Assignment 1: The Airline Reservation Problem

You are to write a program to assign seats on each flight for an airline's only flight with a capacity of 80 seats (20 rows, and 4 columns).

Your program should display the following menu:

Enter 1 for first class, 2 for business class, 3 for economy class, 4 to display the available seats, and 5 to quit:

If a person enters 1, your program should assign a seat in the first class (rows 1 - 3). If a person enters 2, your program should assign a seat in business class (rows 4 - 7). If a person enters 3, your program should assign a seat in economy class (rows 8 - 20). If the seat is available at the given class, then the program should issue a boarding pass with the row and column number, and class type. If a customer asks for a class that is already full, then your program should ask the customer if it is okay to assign a seat in one of the other classes. If the flight is full, the program should print the message "This Flight is Full, Next Flight leaves in Three Hours".

Use a two dimensional array to represent the seating chart of the plane. Initialize all the elements of the array to 0 to indicate that all seats are empty. As each seat is assigned, set the corresponding element of the array to 1 to indicate that the seat is no longer available.

Your program should of course never assign a seat that has already been assigned.

You are required to write this program using functions. The prototype of your functions may look like as below or you may define them in any other way that you wish :

```
void makeFirstClassReservation(int a[][4] );

void makeBusinessClassReservation(int a[][4] );

void makeEconomyClassReservation(int a[][4] );

bool isFirstClassFull(int a[][4] );

bool isBusinessClassFull(int a[][4] );

bool isEconomyClassFull(int a[][4] );

void displayArray( int a[][4]);
```

etc.

If you define proper functions and use them in your main program, then your main program should not be complicated, and it will consist of small number of manageable statements.