

2021학년도 1학기 ICE2013 객체지향프로그래밍응용

강의 소개 및 개요

최학남 (하-401호, 860-8427, xncui@inha.ac.kr)

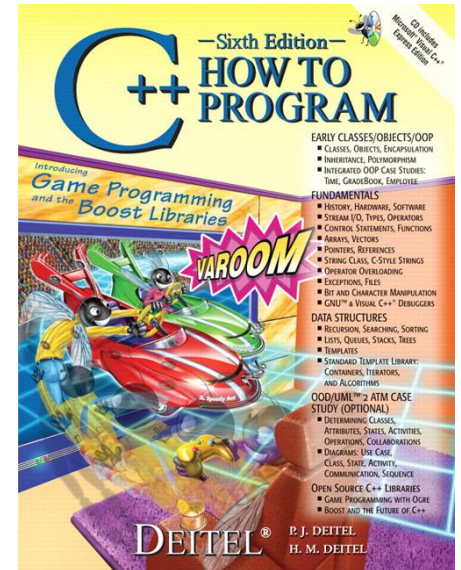
강의 개요

■ 교재

- H. M. Deitel and P. J. Deital, *C++ How to Program*, 6th Edition, Pearson Education International, 2008.

■ 강의 방식

- 이론 : 주당 2시간의 프로그래밍 이론
 - 시각 2회 결석 처리
- 실습 : 주당 2시간의 프로그래밍 실습
 - 실습은 Visual Studio 2017 환경에서 진행
 - 조교 진행



강의 개요

■ 선수과목 및 실습 경험

- 프로그래밍 기초, (필수) 객체지향프로그래밍 1 or 객체지향프로그래밍
- 각종 프로그래밍 경험
 - 多多益善이나, 본 강좌를 통해서 보완할 수 있음
- Computer와 digital data에 대한 이해 필요
 - ICE1001 정보통신입문에서 배운 것들
 - Byte, bit, address, memory, CPU, ...
- Computer 활용 능력
 - Microsoft Windows 10 이상의 운영체제 환경

강의 개요

■ 강의시간 外 의견교환 방법

- Office hour를 이용한 교수 연구실 방문
- I-class (learn.inha.ac.kr)
 - 주요 공지 사항 확인, public Q&A
 - 반드시 하루에 1회 이상 방문할 것
- Email
 - Private Q&A 위주
 - Email 예절 엄수할 것
 - 인사 → 자기소개 → 용건 → 인사

강의의 목적 및 중요성

■ 응용 소프트웨어를 작성할 수 있는 도구 확보

- C++ 언어 기반의 객체 지향의 프로그래밍 능력 확보

■ 향후 교과목과의 연계

- 자료구조, 알고리즘을 통해 컴퓨터를 이용한 문제 분석 및 해결 능력을 향상시킴
- 다양한 응용 교과목의 실제 구현의 기초
- 현대 융합 공학은 S/W와 H/W의 경계가 없음

평가 계획

■ 시험 및 퀴즈

- 중간고사 : 30%, 기말고사 : 30%
- 실습 및 프로그래밍 과제 : 20%
- 퀴즈/프로젝트: 10%
- 출석 및 수업태도 : 10%
- 평가방법
 - 고학년 수강생들은 1학년과 별도로 평가함

■ 성실도 및 도덕성

- 과제 및 시험 부정 행위시 **F**학점 처리 및 학부 통보
- 실습 태도 불량시 출석점수 감점
 - 문자메시지, 카카오톡, 인터넷 서핑, 전화 울림, 졸음 등등

주요 강의 내용

- C++ 클래스(class)를 이용한 객체지향설계 기법
 - Chap. 3: Class 정의
 - Chap. 9: Class 사용
 - Chap. 10: Encapsulation (정보은닉) 및 abstraction (추상화)
 - Chap. 11: Operator overloading (연산자 오버로딩)
 - Chap. 12: Inheritance (상속)
 - Chap. 13: Polymorphism (다형성)
 - Chap. 14: Template 및 활용

Week	Contents
1	Introduction to this course
2	Chapter 3 - Introduction to classes and objects
3	Chapter 3 - Introduction to classes and objects
4	Chapter 9 - Classes: A deeper look 1 (Fri.)
5	Chapter 9 - Classes: A deeper look 1
6	Chapter 10 - Classes: A deeper look 2
7	Chapter 10 - Classes: A deeper look 2
8	Middle exam (paper and practice)
9	Chapter 11 - operator overloading
10	Chapter 11 - operator overloading
11	Chapter 12 - inheritance
12	Chapter 12 - inheritance
13	Chapter 13 - polymorphism
14	Chapter 13 - virtual function
15	Chapter 14 - templates
16	Final exam (paper and practice)



Ch.3 :

3	Introduction to Classes and Objects	81
3.1	Introduction	82
3.2	Classes, Objects, Member Functions and Data Members	82
3.3	Overview of the Chapter Examples	84
3.4	Defining a Class with a Member Function	84
3.5	Defining a Member Function with a Parameter	88
3.6	Data Members, <i>set</i> Functions and <i>get</i> Functions	91
3.7	Initializing Objects with Constructors	98
3.8	Placing a Class in a Separate File for Reusability	102
3.9	Separating Interface from Implementation	106
3.10	Validating Data with <i>set</i> Functions	112

Ch.9 :

9	Classes: A Deeper Look, Part I	487
9.1	Introduction	488
9.2	Time Class Case Study	489
9.3	Class Scope and Accessing Class Members	494
9.4	Separating Interface from Implementation	496
9.5	Access Functions and Utility Functions	498
9.6	Time Class Case Study: Constructors with Default Arguments	500
9.7	Destructors	506
9.8	When Constructors and Destructors Are Called	507
9.9	Time Class Case Study: A Subtle Trap—Returning a Reference to a private Data Member	510
9.10	Default Memberwise Assignment	513



Ch.10 :

10 **Classes: A Deeper Look, Part 2** **530**

10.1	Introduction	531
10.2	const (Constant) Objects and const Member Functions	531
10.3	Composition: Objects as Members of Classes	541
10.4	friend Functions and friend Classes	548
10.5	Using the this Pointer	552
10.6	Dynamic Memory Management with Operators new and delete	557
10.7	static Class Members	559

Ch.11 :

|| Operator Overloading; String and Array Objects **578**

11.1	Introduction	579
11.2	Fundamentals of Operator Overloading	580
11.3	Restrictions on Operator Overloading	581
11.4	Operator Functions as Class Members vs. Global Functions	583
11.5	Overloading Stream Insertion and Stream Extraction Operators	584
11.6	Overloading Unary Operators	588
11.7	Overloading Binary Operators	588
11.8	Case Study: Array Class	589
11.9	Converting between Types	601
11.10	Case Study: String Class	602
11.11	Overloading ++ and --	614
11.12	Case Study: A Date Class	616
11.13	Standard Library Class string	620

Ch.12 :

12 Object-Oriented Programming: Inheritance 640

12.1	Introduction	641
12.2	Base Classes and Derived Classes	642
12.3	protected Members	645
12.4	Relationship between Base Classes and Derived Classes	645
12.4.1	Creating and Using a CommissionEmployee Class	646
12.4.2	Creating a BasePlusCommissionEmployee Class Without Using Inheritance	651
12.4.3	Creating a CommissionEmployee–BasePlusCommissionEmployee Inheritance Hierarchy	657
12.4.4	CommissionEmployee–BasePlusCommissionEmployee Inheritance Hierarchy Using protected Data	662
12.4.5	CommissionEmployee–BasePlusCommissionEmployee Inheritance Hierarchy Using private Data	669
12.5	Constructors and Destructors in Derived Classes	677
12.6	public, protected and private Inheritance	685

Ch.13 :

I3 Object-Oriented Programming: Polymorphism 693

13.1	Introduction	694
13.2	Polymorphism Examples	696
13.3	Relationships Among Objects in an Inheritance Hierarchy	697
13.3.1	Invoking Base-Class Functions from Derived-Class Objects	697
13.3.2	Aiming Derived-Class Pointers at Base-Class Objects	705
13.3.3	Derived-Class Member-Function Calls via Base-Class Pointers	706
13.3.4	Virtual Functions	708
13.3.5	Summary of the Allowed Assignments Between Base-Class and Derived-Class Objects and Pointers	714
13.4	Type Fields and switch Statements	715
13.5	Abstract Classes and Pure virtual Functions	715
13.6	Case Study: Payroll System Using Polymorphism	717
13.6.1	Creating Abstract Base Class Employee	719
13.6.2	Creating Concrete Derived Class SalariedEmployee	722
13.6.3	Creating Concrete Derived Class HourlyEmployee	724
13.6.4	Creating Concrete Derived Class CommissionEmployee	727
13.6.5	Creating Indirect Concrete Derived Class BasePlusCommissionEmployee	729
13.6.6	Demonstrating Polymorphic Processing	731

Ch.14 :

14 Templates

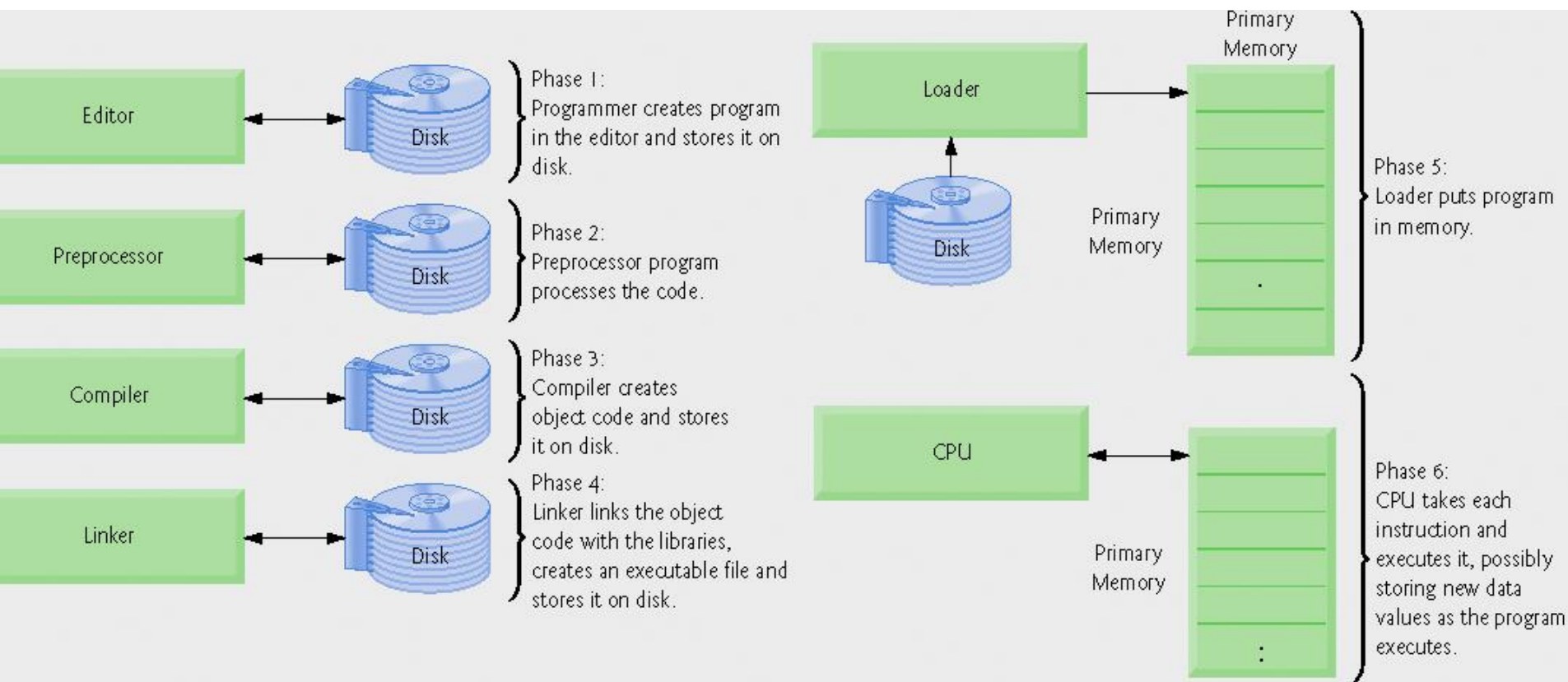
756

14.1	Introduction	757
14.2	Function Templates	758
14.3	Overloading Function Templates	761
14.4	Class Templates	761
14.5	Nontype Parameters and Default Types for Class Templates	768



1 Typical C++ Development Environment

C++ programs normally undergo six phases



Typical C++ environment.

Review (Chapter 2)

A standard “main” program

```
1 // A standard “main” program
2 #include <iostream> // allows program to output and input data
3
4 // function main begins program execution
5 int main()
6 {
7     int number1;
8
9     std::cout << “Welcome to C++\n”; // display message
10    std::cin >> number1;
11
12    return 0;
13
14 } // end function main
```

- Namespace → std::
- Standard output/input stream object
std::cout, std::cin

- Preprocessor directives - <iostream> → no need “;”
- White space, tab, blank line
- Function main → braces
- Statements → “;”

- Escape characters → “\n”
- return statement

Review (Chapter 2)

- **Variable** → variable name, type, declaration, _ or __
- **Concatenating stream and extracting operator**
 - Stream insertion operator <<
 - Stream extraction operator >>
- **Stream manipulator std::endl**
- **Assignment operator =**
- **Arithmetic operators**
+, -, *, /, %
- **Straight-line form**
- **Grouping subexpressions**
- **Rules of operator precedence**
- **Condition** → true, false
- **if statement**
- **Equality and Relational Operators (7)**
- **using declarations**

Data type table

Data type			Byte	Scope
Integer	signed	short	2	-32768~32767
		int	4	-2147483648~2147483647
		long	4	-2147483648~2147483647
		long long	8	-9,223,372,036,854,775,808~9,223,372,036,854,775,807
	unsigned	unsigned short	2	0~65535
		unsigned int	4	0~4294967295
		unsigned long	4	0~4294967295
Character	signed	char	1	-128~127
	unsigned	unsigned char	1	0~255
Floating-point		float	4	-3.4e-38 ~3.4e+38
		double	8	-1.7e-308 ~1.7e+308

Review (Chapter 4)

- Algorithms
- **Sequence structure**
 - Programs executed sequentially by default
- **Selection structures**
 - `if`, `if...else`, Nested `if...else` statements, `switch`
 - `?` :
- **Repetition structures**
 - `while`, `do...while`, `for`
- C & C++ Keywords
- Assignment Operators
- Increment and Decrement Operators
 - Preincrement, Postincrement

Review (Chapter 5)

- **for** repetition statement
- **do ... while** repetition statement
- **switch** multiple-selection statement

Review (Chapter 6)

Function definition:

→ return type and parameter list (argument list), function prototype and function header

The usage of Standard Library:

→ rand(), srand(), ctime(), header: cstdlib.h, ctime.h

Review (Chapter 7)

1. What is array

- data structure
- large amounts of same type data

2. Array declaration

- multi-dimensional declaration

3. Manipulate arrays

- static array vs. auto array; constant variable

4. Passing arrays to function

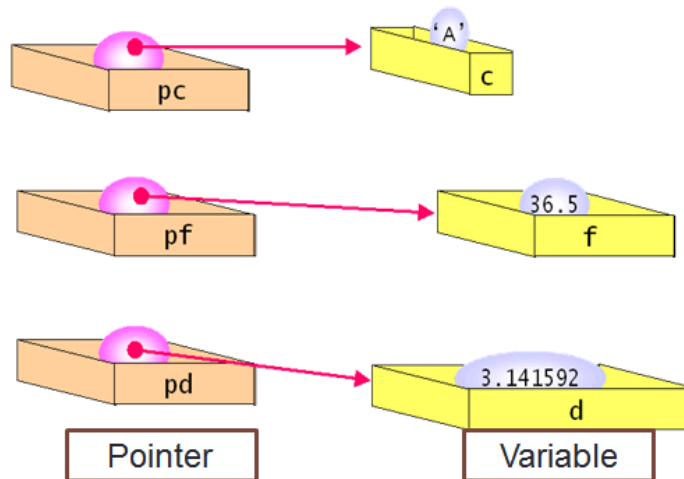
- By reference and by value

5. Searching arrays with linear search

6. Sorting arrays: insertion sort

Review (Chapter 8)

1. **Pointer variable declarations and initialization**
2. **Pointer operators**
 - * and &
 - `int y = 5; int *yPtr; yPtr = &y;`
3. **Passing arguments to functions**
4. **Pointer arithmetic**



프로그래밍을 잘 하기 위해...

- 프로그래밍에 대해 긍정적인 마인드를 가지세요
- 반복 연습을 하세요
- 절대 중도에 포기하지 마세요 (Never give up!)
- 잘 하는 동료들의 어깨너머로 배우세요
- 좋은 책을 골라 계속 참고하세요
- 인터넷 레퍼런스 (예: MSDN)을 적극 활용하세요
- 꼭 해내야 하는 프로젝트 참여 기회를 만드세요

Student Honor Code

- **인하대학교 학생으로서 또한 본 교과목의 수강생으로서 본인은,**
 - 시험 및 성적에 관련된 어떠한 부정 행위에도 연루되지 않을 것이다.
 - 시험이나 과제에 관련된 금지된 정보나 자료를 입수하거나 유출하지 않을 것이다.
 - 인터넷의 익명 자료를 포함하여 타인의 결과물을 표절하지 않을 것이다.
 - 타인을 위해 시험 및 과제 등을 대리로 작성하지 않을 것이다.
 - 허가 없이 본 교과목에 관련된 제출물을 타 교과목을 위해 제출하지 않을 것이다.
 - 부과되는 과제는 성실히 본인의 힘으로 해결하여 제출할 것이다.
 - 다른 학생의 학습에 방해되는 행동을 하지 않을 것이다.
 - 기타 나의 양심에 反하는 어떠한 행위도 하지 않을 것이다.