


2025-2학기

UNIFIED
MODELING
LANGUAGE™



객체 지향 설계와 분석을 위한
UML
기초와 응용



한빛아카데미



소프트웨어 공학
이론과 실제

한빛아카데미

게임소프트웨어공학(GSE)

[2주차-2] 실습 1장 UML의 이해

1. UML 용도와 특징

2. 객체지향 모델링

3. 과제/진도/Q&A

 한국공학대학교

이 강의자료는 여러분을 위한 담당교수의 경험/생각과 한빛아카데미에서 제공한 교재 자료를 활용해서 작성하였습니다.

0

[실습 1장] UML의 이해

■ 주요 내용

1. UML 용도와 특징


2. 객체지향 모델링

■ 학습 목표

1. UML의 개념과 특징 이해

2. 객체지향 개념과 특징 이해

3. 모델링 방법 이해

 한국공학대학교

실습 1장 UML의 이해

1



실습교재

1장

UML의 이해



1. UML 용도와 특징

2. 객체지향 모델링

3. 과제/진도/Q&A



실습 1장 UML의 이해


2

1. UML의 용도와 특징

■ UML의 탄생과 특징

- UML(Unified Modeling Language, **통합 모델링(모형화) 언어**)
 - 시스템 개발을 위한 시각적인 **설계 표기법 제공**
 - 객체지향 시스템을 개발할 때 **산출물의 명세화, 시각화, 문서화에 사용**
 - 개발 시스템을 이해하기 쉬운 형태로 표현하여 분석가, 설계자, 의뢰인(고객, 사용자)의 **효율적인 의사소통에 기여**

➔ **UML은 표준화된 통합 모델링 언어!**




실습 1장 UML의 이해

3


1. UML의 용도와 특징

■ UML의 탄생과 특징


- 그래디 부치 Grady Booch, 제임스 럼버 James Rumbaugh, 이바 야콥슨 Ivar Jacobson이 UML 초안 연구
- 1997년 OMG Object Management Group, 객체관리그룹)에서 여러가지 객체지향 표기법들을 통합하여 UML을 발표



그래디 부치




이바 야콥슨




제임스 럼버

1994년 UML 초안 연구



OBJECT MANAGEMENT GROUP®
1997년 UML 발표

그림 1-1 UML을 개발한 사람들



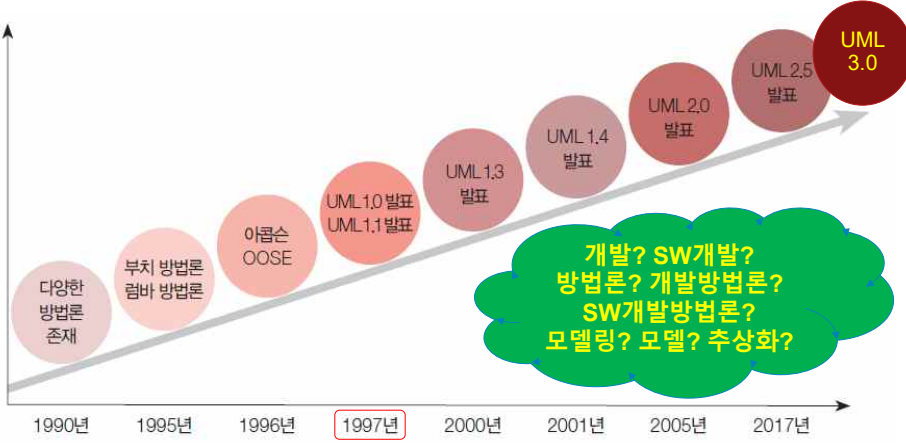
한국과학기술대학교

실습 1장 UML의 이해

4

1. UML의 용도와 특징

■ UML의 탄생과 특징



다양한 방법론 존재

1990년

부치 방법론
럼바 방법론

1995년

아콥슨
OOSE

1996년

UML 1.0 발표
UML 1.1 발표

1997년

UML 1.3 발표

2000년

UML 1.4 발표

2001년

UML 2.0 발표

2005년


UML 2.5 발표

2017년

UML 3.0

개발? SW개발?
방법론? 개발방법론?
SW개발방법론?
모델링? 모델? 추상화?

그림 1-2 UML의 발전 과정



한국과학기술대학교

실습 1장 UML의 이해

5


Copyrighted by JinWeonSuk(2025-2학기)

3

1. UML의 용도와 특징

■ UML이 제공하는 표준화된 다이어그램(9가지)

- 유스케이스 다이어그램Use-case Diagram
- 클래스 다이어그램Class Diagram
- 순차 다이어그램Sequence Diagram
- 통신 다이어그램Communication Diagram
- 활동 다이어그램Activity Diagram
- 상태 다이어그램State Diagram
- 컴포넌트 다이어그램Component Diagram
- 배치 다이어그램Deployment Diagram
- 패키지 다이어그램Package Diagram

 한국공학대학교


실습 1장 UML의 이해

6

1. UML의 용도와 특징

■ UML의 특징

- UML은 시각화Visualization 언어
 - 소프트웨어의 개념 모델을 시각적인 형태로 표현, 명확히 정의된 표준화된 다이어그램을 제공
 - 작성된 다이어그램을 이용해서 오류가 없는 원활한 의사소통 가능
- UML은 명세화Specification 언어
 - 소프트웨어 개발과정인 분석, 설계 단계의 과정에서 필요한 모델을 정확하고 완전하게 명세화(Specification)해서 만들 수 있음
 - 명세화된 다이어그램의 기호들은 의미를 담고 있으며, 추상적이지만 고유의 특성을 갖고 있음
- UML은 구축Construction 언어
 - 자바Java, C++, 비주얼 베이직Visual Basic, C# 같은 다양한 프로그래밍 언어로 표현 (생성) 가능
 - UML로 설계된 모델을 프로그램 코드로 자동 변환 가능, 이미 구축된 소스 코드는 UML로 역변환해서 분석하는 역공학Reverse Engineering도 가능
- UML은 문서화Documentation 언어
 - StarUML, 투게더Together 등의 개발도구CASE Tool들을 이용해서 설계한 내용을 자동으로 문서화 가능

 한국공학대학교

실습 1장 UML의 이해

7

1. UML의 용도와 특징

UML과 모델링

```
#include <stdio.h>
int main() {
    int a, b;
    scanf("%d %d", &a, &b);
    printf("a + b = %d", a + b);
}
```

(a) C로 구현한 덧셈 프로그램

```
def add():
    num = int(input("num1:"))
    num2 = int(input("num2:"))
    result = num + num2
    print("두수의 합은 " result)
```

(c) 파이썬으로 구현한 덧셈 프로그램

```
import java.util.Scanner;
public class AddDemo {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in );
        int a = in.nextInt();
        int b = in.nextInt();
        System.out .printf("%d + %d = %d", a, b, a+b);
    }
}
```

(b) 자바로 구현한 덧셈 프로그램

모델링은 개발하려는 프로그램을 시각적으로 표현하는 것이며, 이때, 외뢰자의 요구에 맞게 쉽게 수정해서 결과적으로는 유지보수 시간을 줄여 생산성을 높일 수 있음

(a) 덧셈 연산 프로그램의 시각적 표현

(b) 사칙연산 프로그램의 시각적 표현

그림 1-3 다양한 언어로 구현한 덧셈 프로그램

그림 1-4 프로그램의 시각적 표현

1. UML의 용도와 특징

모델링이 필요한 이유

모델링이 꼭 필요하지는 않다.

(a) 개인이 제어할 수 있는 작업 : 개집 짓기

모델링이 필요하다.

(b) 개인이 제어할 수 없는 작업 : 큰 건축물 짓기

그림 1-5 모델링이 필요한 이유

Copyrighted by JinWeonSuk(2025-2학기)

5



UNIFIED
MODELING
LANGUAGE™

실습교재

1장

UML의 이해



1. UML 용도와 특징

2. 객체지향 모델링

3. 과제/진도/Q&A



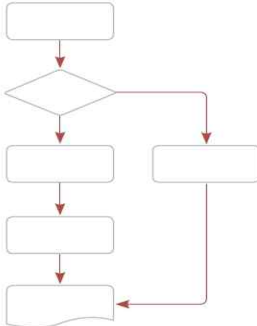
한국외국대학교

실습 1장 UML의 이해

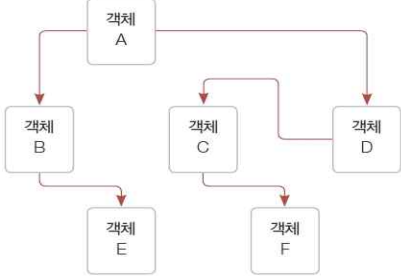
10

2. 객체지향 모델링: 객체지향 개념

- 객체지향의 개념
 - C++, 자바 등의 언어를 이용해서 자료구조를 중심으로 객체를 설계한 후, 이들의 흐름을 설계하는 방식
 - 기본 요소: 객체Object, 클래스Class, 메시지Message




(a) 절차 지향 방법



(b) 객체 지향 방법

그림 1-6 절차 지향 방법과 객체 지향 방법



한국외국대학교

실습 1장 UML의 이해

11

2. 객체지향 모델링

■ 객체지향의 개념

- 객체와 클래스
 - 객체: 현실에 존재하는 모든 것(구체적)
 - 클래스: 개념적으로 객체를 생성할 수 있는 틀(개념적), 클래스는 그 자체만으로는 사용할 수 없고, 객체로 생성되어져서 사용됨



그림 1-7 클래스와 객체의 관계



붕어빵 기계 = 클래스 붕어빵 = 객체
그림 1-8 클래스와 객체의 예 : 붕어빵 기계와 붕어빵

2. 객체지향 모델링

■ 객체지향의 개념

- 객체와 클래스 예
 - 아래한글에서 실행 아이콘 = 클래스, 빈 문서 1, 2, 3, 4, 5 등 = 객체
 - 실행 아이콘만으로는 문서 작성 불가

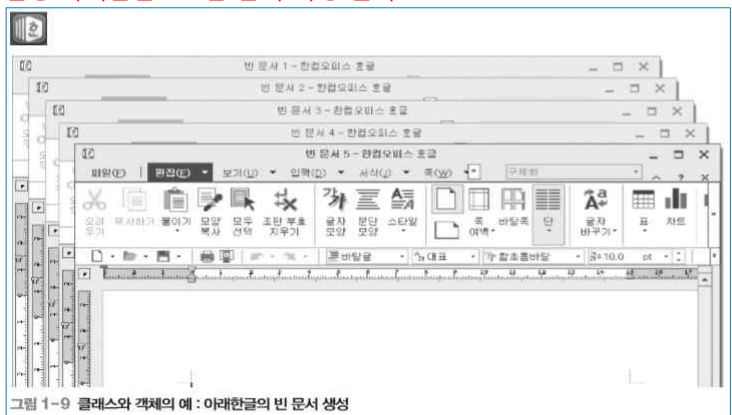


그림 1-9 클래스와 객체의 예 : 아래한글의 빈 문서 생성

2. 객체지향 모델링

■ 객체지향의 개념

- 객체(실체)와 클래스(들)

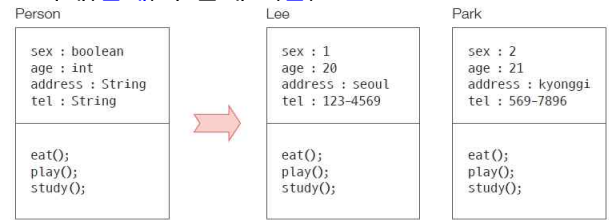


그림 1-10 프로그램에서 클래스와 객체의 예

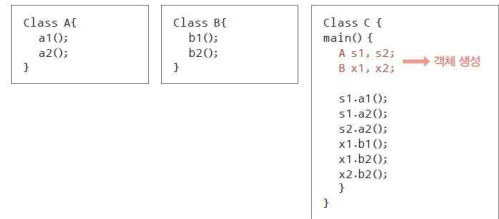


그림 1-11 자바 프로그램에서 객체 생성

2. 객체지향 모델링

■ 객체지향의 개념

- 메시지
 - 객체 간의 상호작용 수단
 - 신호: 하나의 객체가 다른 객체에 특정 작업을 요청하는 것(메시지)
 - 송신 객체_{Sender}: 메시지를 보내는 객체
 - 수신 객체_{Receiver}: 메시지를 받아서 동작을 수행하는 객체

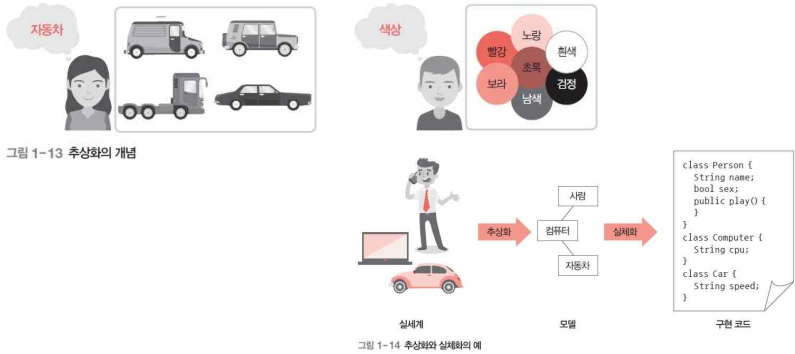


그림 1-12 메시지 전달

2. 객체지향 모델링: 객체지향 특징

■ 객체지향의 특징

- 추상화
 - 특정 측면을 강조하여 나타내는 것
 - 객체지향은 클래스를 이용하여 실세계에 대응하는 추상 모델을 만듦
 - 실체화: 추상화한 모델링을 프로그램으로 구현



2. 객체지향 모델링

■ 객체지향의 특징

- 캡슐화
 - 데이터와 처리담당 **오퍼레이션**이 한 틀 안에서 결합되어 객체라는 단위로 묶여 사용되는 것
 - 캡슐화를 통해 **정보 은닉** Information Hiding 가능
 - 정보은닉을 통해 보다 높은 독립성, 유지보수성, 향상된 이식성을 제공

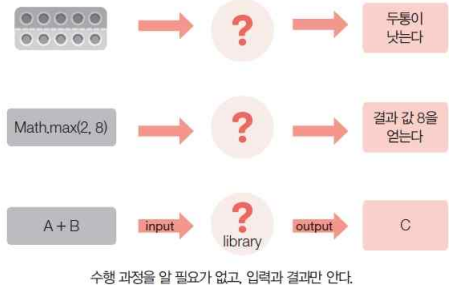


그림 1-15 캡슐화의 개념

2. 객체지향 모델링

■ 객체지향의 특징

- 상속
 - 프로그램을 쉽게 확장할 수 있도록 도와주는 수단
 - 객체지향 패러다임에서만 구현 가능
 - 정보를 공개하고 재사용하는 개념
 - **상세화**: 상위 클래스 속성을 상속받아 하위 클래스에서 실체화 하는 관계
 - **일반화**: 하위 클래스의 공통 특성을 추상화해서 상위 클래스로 정의하는 것



그림 1-16 일반화와 상세화로 표현된 상속

2. 객체지향 모델링

■ 객체지향의 특징

- 상속의 예
 - 자바 언어로 상속 표현할 때, **extends** 키워드 사용

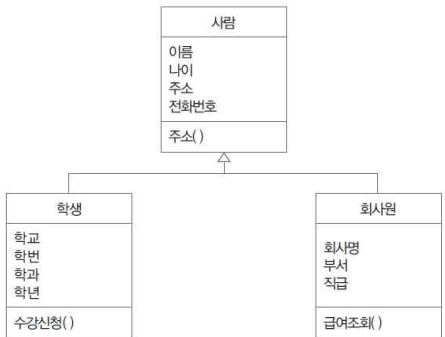
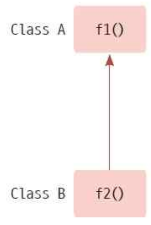


그림 1-17 상속을 표현한 클래스 다이어그램

```
class A {  
    f1();  
}  
class B extends A {  
    f2();  
}  
  
main() {  
    B x;  
    x.f2();  
    x.f1();  
}
```

그림 1-18 자바로 구현한 상속 예



2. 객체지향 모델링

■ 객체지향의 특징

- 다형성
 - 여러 클래스에 같은 이름의 함수가 존재하지만, **동작은 다르게 수행하는 것**
 - 하위 클래스에서는 그들만의 **고유한 속성과 오퍼레이션 재정의의 필요**
 - 객체지향 언어에서 **메서드 오버라이딩** Method Overriding 방식으로 구현

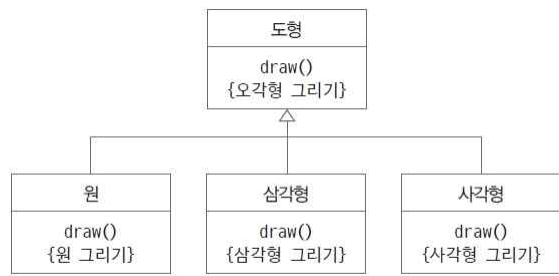
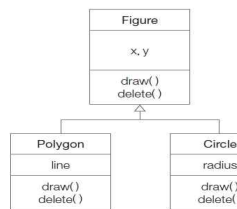


그림 1-19 다형성의 예

2. 객체지향 모델링: 추상 클래스와 인터페이스

■ 추상 클래스와 인터페이스

- 추상 클래스
 - 클래스의 명칭과 메서드는 있으나 **메서드의 처리 내용은 없는 클래스**
 - 상속을 통해서 메서드가 **구현** Implementation
 - 추상 메서드 외에 **일반적인 속성과 메서드**를 가질 수 있음
 - 메서드의 다형성을 지원



```
abstract class Figure {
    public void setCoo(point x, point y) {
        my_X = x;
        my_Y = y;
    }
    public abstract void draw();
    public abstract void delete();
}

class Polygon extends Figure {
    int line;
    public void draw() {.....}
    public void delete() {.....}
}

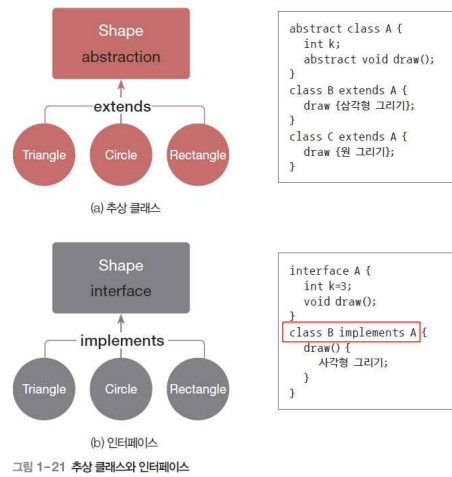
class Circle extends Figure {
    int radius;
    public void draw() {.....}
    public void delete() {.....}
}
```

그림 1-20 추상 클래스의 예

2. 객체지향 모델링

추상 클래스와 인터페이스

- 인터페이스
 - 상수와 추상 메서드만 가짐
 - 여러 개의 인터페이스로부터 상속을 받을 수 있으므로 다중 상속 기능 제공
 - 추상 클래스와 다르게 속성을 가질 수 없으며, 메서드의 구현도 정의할 수 없음



2. 객체지향 모델링: 모델링 개념

모델링 개념

- 모델링
 - 시스템을 구축할 때, 개발자가 고민하고 결정하는 모든 활동
 - 구현 단계 이전의 요구사항 정의, 분석, 설계에서 수행하는 활동
 - 모델(Model, 모형): 모델링의 결과물
 - CASE 도구(Tool): 모델링을 전문적으로 지원하는 툴
 - Ex) StarUML, 로즈Rosey, 투게더Together 등
 - 모델링과 프로그래밍의 비교

표 1-1 모델링과 프로그래밍

구분	모델링	프로그래밍
목적	구축할 시스템의 모습 정의	시스템의 실제 구현
세부 수행 활동	요구 사항 정의, 분석, 설계	소스 코드 편집, 컴파일, 시험, 디버깅
결과물	모델	소스 코드를 포함한 구현된 시스템
표기법	모델링 언어(UML, ERD, DFD)	프로그래밍 언어(자바, C++)
지원 툴	CASE 툴(StarUML, 로즈Rosey, 투게더Together)	개발 툴(Jbuilder, Visual Studio, .Net)

2. 객체지향 모델링: 모델링 방법

■ 모델링 방법

- 부츠의 방법론Booch Method
 - 설계 중심의 방법론으로 시스템을 몇 개의 뷰View로 분석할 수 있다고 보고, 뷰를 모델 다이어그램으로 표현
 - 거시적 개발 프로세스와 미시적 개발 프로세스를 모두 포함하고, 단계적 접근과 자동화 툴을 지원
 - 객체지향 방법론에 대한 광범위한 이론적 배경을 제시하여, 더 넓게 바라볼 수 있는 안목을 제공
- 야콥슨의 OOSE EObject-Oriented Software Engineering
 - 유스케이스를 강조한 방법론
 - 유스케이스는 외부 행위자와 상호작용하는 시스템의 요구사항을 정의하고, 정의된 유스케이스는 개발, 테스트, 검증 단계에서 중요하게 사용
 - 방법론이 복잡하여 초보자에게는 어렵지만, 큰 규모의 시스템을 개발할 때 효율적

2. 객체지향 모델링

■ 모델링 방법

- 럼바의 OMT Object-Modeling Technique
 - 하나의 시스템 기술을 위해서 객체 모델(Object Model), 동적 모델(Dynamic Model), 기능 모델(Functional Model)의 3가지 모델 사용
 - 시스템 분석에서 3가지 모델을 이용해서 시스템이 요구하는 객체 기술
 - 객체(정보) 모델: 시스템에서 필요한 모델을 찾아내고, 객체의 속성과 객체 간의 관계를 규명
 - 동적 모델: 객체 모델에서 나타난 객체들의 행위와 상태를 포함하는 생명주기를 나타냄
 - 기능 모델: 각 객체의 변화로 인해 다른 상태로 전이가 되었을 때 수행되는 동작들을 기술
- UML
 - 부츠 방법론, OOSE 및 OMT를 하나로 합한 방법론
 - 분산 객체Distributed Objects의 표준이 되었고, OMG에서 CORBA Common Object Request Broker Architecture의 표준 분석설계방법론으로 채택



실습교재

1장

UML의 이해



1. UML 용도와 특징

2. 객체지향 모델링

3. 과제/진도/Q&A




실습 1장 UML의 이해

26

3. 과제/진도: [과제#1] 이론1장+실습1장 연습문제 풀이

▪ [과제#1] 이론1장+실습1장 연습문제 풀이(2점)

- 작성 방법: 이론1장/실습1장의 연습문제(전체)를 풀이하여 작성방법 적용하여 작성
- 제출 방법: 제출파일명 부여방법 적용하여 PDF 파일로 변환하여 LMS의 과제란에 제출기한 이내에 제출
- 내용/분량: 제한 없음
- 제출 파일: [과제#1]-이론1장+실습1장연습문제-"게임공학과"-**"학번"**-**"이름"**-20250918
- 예시) [과제#1]-이론1장+실습1장연습문제-게임공학과-2023123456-한게임-20250918
- 제출 기한: 2025.9.18. (목요일), 24시 이전까지



실습 1장 UML의 이해

27

3. 과제/진도: 2주차/전체			
주차	강의 내용	수업 유형	학습 활동
1	총론4장 소프트웨어+이론1장 SE개요+Q&A 과목 사전 일문지 작성	대면수업(이론/실습)	대면수업/실습, 과제 해결
2	이론2장 SW의 품질+실습1장 UML 이해+실습12장(+@) starUML 모델링도구 사용법/설치/피드백	대면수업(이론/실습)	대면수업/실습, 과제 해결
3	이론3장 SW 개발프로세스+실습2장 UML 구성요소/뷰+프로젝트 팀원성/주제 선정/피드백	대면수업(이론/실습)	대면수업/실습, 과제 해결
4	이론4장 SW 개발방법론(DevOps+UP)+실습3장 유스케이스 다이어그램+문제기술서(SOP) 작성/피드백	대면수업(이론/실습)	대면수업/실습, 과제 해결
5	국경일(개천절) 휴강(15주차 보장)	국경일 휴강	국경일 휴강
6	이론5장 프로젝트 관리+실습4장 클래스 다이어그램+프로젝트정의서(PC) 작성/피드백	대면수업(이론/실습)	대면수업/실습, 과제 해결
7	이론6장 SW 비용산정+실습5장 순차 다이어그램+프로젝트관리계획서(PMP) 작성/피드백	대면수업(이론/실습)	대면수업/실습, 과제 해결
8	이론7장 요구사항 도출+실습6장 통신 다이어그램+요구사항정의서(SRD)/중간발표(PT+PMR) 작성/피드백	대면수업(이론/실습)	대면수업/실습, 과제 해결
9	중간고사(필기+개인)+프로젝트 중간발표(PT+PMR+팀별)/피드백	대면수업(시험/발표)	서술형 필기시험/구두발표
10	이론8장 객체지향 분석+실습7장 활동 다이어그램+요구사항추적표(RTM) 작성/피드백	대면수업(이론/실습)	대면수업/실습, 과제 해결
11	이론9장 모듈화 설계+실습 8장 상태 다이어그램+1. 요구사항명세서(SRS) 작성/피드백	대면수업(이론/실습)	대면수업/실습, 과제 해결
12	이론10장 설계 패턴+이론11장 객체지향 설계+실습9장 컴포넌트 다이어그램+설계기술서(SDD) 작성/피드백	대면수업(이론/실습)	대면수업/실습, 과제 해결
13	이론12장 인스펙션+이론13장 코딩+실습10장 배치 다이어그램+구현계획서(SIP) 작성/피드백	대면수업(이론/실습)	대면수업/실습, 과제 해결
14	이론14장 화이트박스 테스트+이론15장 블랙박스 테스트+실습 11장 패키지 다이어그램+시험계획서(STP)/시험설계서(STD) 작성/피드백	대면수업(이론/실습)	대면수업/실습, 과제 해결
15	이론16장 SW 개발 적용 기술+실습12장 깃과 깃허브 활용 방법+구현결과서(SIR)/시험결과서(STR)/최종발표(PT+PCR) 작성/피드백	대면수업(이론/실습) (5주차 보장)	대면수업/실습, 과제 해결 (5주차 보장)
16	기말고사(L&L+개인중간고사(필기+개인)+프로젝트 중간발표(PT+PCR+팀별)/피드백+최종보고서(PCR) 제출	대면수업(시험/발표)	서술형 필기시험/구두발표
한국광명대학교 실습 1장 UML의 이해 28			

3. 과제/진도: 2주차(결과)-3주차(계획)					
2주차 결과/3주차 계획					
주차	주요학습내용	학습성과 학습목표	수업운영방법	학습준비사항	교재, 참고도서 (page)
2주차	<ul style="list-style-type: none">이론 2장 SW의 품질실습 1장 UML의 이해실습 12장 StarUML을 이용한 SW 개발방법	<ol style="list-style-type: none">SW 품질의 이해UML의 이해StarUML 모델링 도구사용법 이해	<ul style="list-style-type: none">대면강의+실습[과제#1] 이론(1장)+실습(1장) 연습문제 풀이	교재 준비(이론, 실습) 및 이론 2장/실습 1장, 12장 읽어 보기	강의계획서+이론/실습 교재/참고도서+강의자료
3주차	<ul style="list-style-type: none">이론 2장 SW의 품질실습 2장 UML 구성요소와 뷰실습 12장 StarUML을 이용한 SW 개발방법2	<ol style="list-style-type: none">SW 품질의 이해UML 구성요소와 뷰의 이해StarUML 모델링 도구의 사용법 이해2	<ul style="list-style-type: none">대면강의+실습[과제#2] 이론(2장)+실습(2장) 연습문제 풀이	교재 준비(이론, 실습) 및 이론 2장/실습 2+12장 읽어 보기	강의계획서+이론/실습 교재+참고도서+보충자료
한국광명대학교 실습 1장 UML의 이해 29					

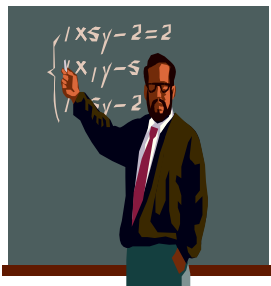
3. 과제/진도: 3주차 안내

- 강의계획서는 **잘(정확히) 숙지**하고, 매주 강의 **진도 확인**하기
- [과제#1] 이론1장+실습1장 연습문제 풀이(개인별, 2점)**
 - 제출 파일: [과제#1]-이론1장+실습1장연습문제-"게임공학과"-**"학번"**-**"이름"**-20250918
예시) [과제#1]-이론1장+실습1장연습문제-게임공학과-2023123456-한게임-20250918
 - 제출 기한: 2025.9.18. (목요일), 24시 이전까지
- 강의교재(이론/실습) 준비 및 교재 1~2장 읽어보기

 **3주차: 이론 2장/실습 2, 12장+@ 강의 및 실습**

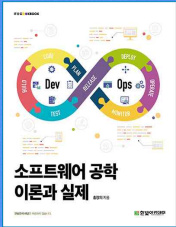
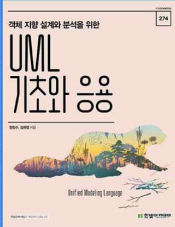

□ 궁금하면 질문하자!

- LMS의 질의응답, 댓글, 쪽지 및 메일 기능을 적극 활용하기 바랍니다!**



학습(學習)은 배우고 익히라는 것이다.
배우기만 힘쓰면 스스로 할 수 없는 사람이 된다!
항상 배우고 익혀야 한다!


Q & A



[GSE 2주차-2] 수고했습니다!

다음 시간에 만납시다~~~

이 과목에서 사용되는 일부 자료, 영상물 등은 강의 내용을 보충하기 위해 교육 목적으로 활용하였습니다. 이 과목의 강의 자료 및 영상물의 불법적 이용, 무단 전재·배포는 법적으로 금지되어 있으니, 학생 여러분은 학습 외 용도의 사용을 주의하시기 바랍니다.

한국과학기술대학교

실습 1장 UML의 이해

32