

대분류/20
정보통신

중분류/01
정보기술

소분류/02
정보기술개발

세분류/02
응용SW엔지니어링

능력단위/30

NCS학습모듈

프로그래밍 언어 응용

LM2001020230_19v4



교육부

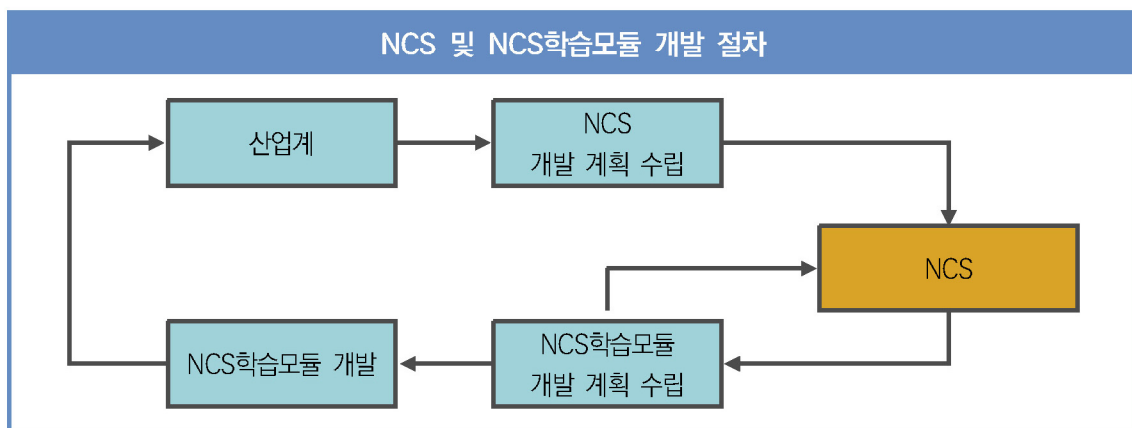
NCS 학습모듈은 교육훈련기관에서 출처를 명시하고 교육적 목적으로 활용할 수 있습니다. 다만 NCS 학습모듈에는 국가(교육부)가 저작권 일체를 보유하지 않은 저작물들(출처가 표기되어 있는 도표, 사진, 삽화, 도면 등)이 포함되어 있으므로 이러한 저작물들의 변형, 복제, 공연, 배포, 공중 송신 등과 이러한 저작물들을 활용한 2차 저작물의 생성을 위해서는 반드시 원작자의 동의를 받아야 합니다.

NCS학습모듈의 이해

※ 본 NCS학습모듈은 「NCS 국가직무능력표준」사이트(<http://www.ncs.go.kr>) 에서 확인 및 다운로드할 수 있습니다.

I NCS학습모듈이란?

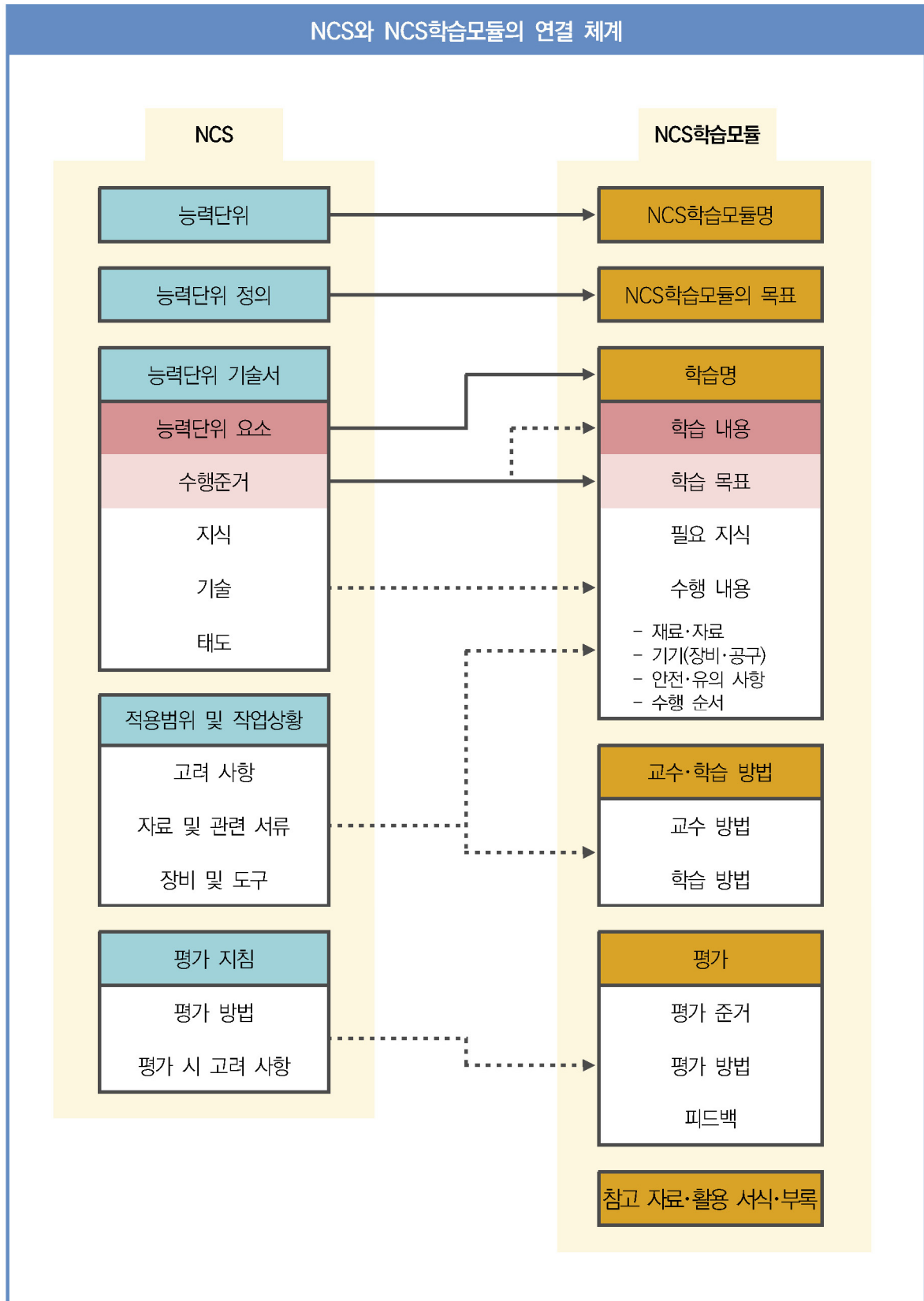
- 국가직무능력표준(NCS: National Competency Standards)이란 산업현장에서 직무를 수행하기 위해 요구되는 지식·기술·소양 등의 내용을 국가가 산업부문별·수준별로 체계화한 것으로 산업현장의 직무를 성공적으로 수행하기 위해 필요한 능력(지식, 기술, 태도)을 국가적 차원에서 표준화한 것을 의미합니다.
- 국가직무능력표준(이하 NCS)이 현장의 ‘직무 요구서’라고 한다면, **NCS학습모듈은 NCS의 능력단위를 교육훈련에서 학습할 수 있도록 구성한 ‘교수·학습 자료’입니다.** NCS학습모듈은 구체적 직무를 학습할 수 있도록 이론 및 실습과 관련된 내용을 상세하게 제시하고 있습니다.



○ NCS학습모듈은 다음과 같은 특징을 가지고 있습니다.

- 첫째, NCS학습모듈은 산업계에서 요구하는 직무능력을 교육훈련 현장에 활용할 수 있도록 성취목표와 학습의 방향을 명확히 제시하는 가이드라인의 역할을 합니다.
- 둘째, NCS학습모듈은 특성화고, 마이스터고, 전문대학, 4년제 대학교의 교육기관 및 훈련기관, 직장교육기관 등에서 표준교재로 활용할 수 있으며 교육과정 개편 시에도 유용하게 참고할 수 있습니다.

○ NCS와 NCS학습모듈 간의 연결 체계를 살펴보면 아래 그림과 같습니다.



II NCS학습모듈의 체계

○ NCS학습모듈은 1. NCS학습모듈의 위치, 2. NCS학습모듈의 개요, 3. NCS학습모듈의 내용 체계, 4. 참고 자료, 5. 활용서식/부록으로 구성되어 있습니다.

1. NCS학습모듈의 위치

○ NCS학습모듈의 위치는 NCS 분류 체계에서 해당 학습모듈이 어디에 위치하는지를 한 눈에 볼 수 있도록 그림으로 제시한 것입니다.

[NCS-학습모듈의 위치]		
대분류	문화·예술·디자인·방송	
중분류	문화콘텐츠	
소분류	문화콘텐츠제작	
세분류		
방송콘텐츠제작	능력단위	학습모듈명
영화콘텐츠제작	프로그램 기획	프로그램 기획
음악콘텐츠제작	아이템 선정	아이템 선정
광고콘텐츠제작	자료 조사	자료 조사
게임콘텐츠제작	프로그램 구성	프로그램 구성
애니메이션 콘텐츠제작	캐스팅	캐스팅
만화콘텐츠제작	제작계획	제작계획
캐릭터제작	방송 미술 준비	방송 미술 준비
스마트문화앱 콘텐츠제작	방송 리허설	방송 리허설
영사	야외촬영	야외촬영
	스튜디오 제작	스튜디오 제작

학습모듈은

NCS 능력단위 1개당 1개의 학습모듈 개발을 원칙으로 합니다. 그러나 필요에 따라 고용단위 및 교과단위를 고려하여 능력단위 몇 개를 묶어 1개 학습모듈로 개발할 수 있으며, NCS 능력단위 1개를 여러 개의 학습모듈로 나누어 개발할 수도 있습니다.

2. NCS학습모듈의 개요

○ NCS학습모듈의 개요는 학습모듈이 포함하고 있는 내용을 개략적으로 설명한 것으로

학습모듈의 목표, 선수학습, 학습모듈의 내용 체계, 핵심 용어 로 구성되어 있습니다.

학습모듈의 목표	해당 NCS 능력단위의 정의를 토대로 학습 목표를 작성한 것입니다.
선수학습	해당 학습모듈에 대한 효과적인 교수·학습을 위하여 사전에 이수해야 하는 학습모듈, 학습 내용, 관련 교과목 등을 기술한 것입니다.
학습모듈의 내용 체계	해당 NCS 능력단위요소가 학습모듈에서 구조화된 체계를 제시한 것입니다.
핵심 용어	해당 학습모듈의 학습 내용, 수행 내용, 설비·기자재 등 가운데 핵심적인 용어를 제시한 것입니다.

제작계획 학습모듈의 개요

학습모듈의 목표

본격적인 촬영을 준비하는 단계로서, 촬영 대본을 확정하고 제작 스태프를 조직하며 촬영 장비와 촬영 소품을 준비할 수 있다.

선수학습

제작 준비(LM0803020105_13v1), 섭외 및 제작스태프 구성(LM0803020104_13v1), 촬영 제작(LM0803020106_13v1), 촬영 장비 준비(LM0803040204_13v1.4), 미술 디자인 협의하기(LM0803040203_13v1.4)

학습모듈의 내용체계

학습	학습 내용	NCS 능력단위 요소	
		코드번호	요소 명칭
1. 촬영 대본 확정하기	1-1. 촬영 구성안 검토와 수정	0803020114_16.3.1	촬영 대본 확정하기
2. 제작 스태프 조직하기	2-1. 기술 스태프 조직 2-2. 미술 스태프 조직 2-3. 전문 스태프 조직	0803020114_16.3.2	제작 스태프 조직하기
3. 촬영 장비 계획하기	3-1. 촬영 장비 점검 과 준비	0803020114_16.3.3	촬영 장비 계획하기
4. 촬영 소품 계획하기	4-1. 촬영 소품 목록 작성 4-2. 촬영 소품 제작 의뢰	0803020114_16.3.4	촬영 소품 계획하기

핵심 용어

촬영 구성안, 제작 스태프, 촬영 장비, 촬영 소품

학습모듈의 목표는

학습자가 해당 학습모듈을 통해 성취해야 할 목표를 제시한 것으로, 교수자는 학습자가 학습모듈의 전체적인 내용흐름을 파악하도록 지도할 수 있습니다.

선수학습은

교수자 또는 학습자가 해당 학습모듈을 교수·학습하기 이전에 이수해야 하는 교과목 또는 학습모듈(NCS 능력단위) 등을 표기한 것입니다. 따라서 교수자는 학습자가 개별 학습, 자기 주도 학습, 방과 후 활동 등 다양한 방법을 통해 이수할 수 있도록 지도하는 것을 권장합니다.

핵심 용어는

해당 학습모듈을 대표하는 주요 용어입니다. 학습자가 해당 학습모듈을 통해 학습하고 평가받게 될 주요 내용을 알 수 있습니다. 「NCS 국가직무능력표준」 사이트(www.ncs.go.kr)의 색인(찾아보기) 중 하나로 이용할 수 있습니다.

3. NCS학습모듈의 내용 체계

○ NCS학습모듈의 내용은 크게 **학습**, **학습 내용**, **교수·학습 방법**, **평가**로 구성되어 있습니다.

학습	해당 NCS 능력단위요소 명칭을 사용하여 제시한 것입니다. 학습은 크게 학습 내용, 교수·학습 방법, 평가로 구성되며 해당 NCS 능력단위의 능력단위 요소별 지식, 기술, 태도 등을 토대로 내용을 제시한 것입니다.
학습 내용	학습 내용은 학습 목표, 필요 지식, 수행 내용으로 구성되며, 수행 내용은 재료·자료, 기기(장비·공구), 안전·유의 사항, 수행 순서, 수행 tip으로 구성된 것입니다. 학습모듈의 학습 내용은 실제 산업현장에서 이루어지는 업무활동을 표준화된 프로세스에 기반하여 다양한 방식으로 반영한 것입니다.
교수·학습 방법	학습 목표를 성취하기 위한 교수자와 학습자 간, 학습자와 학습자 간 상호 작용이 활발하게 일어날 수 있도록 교수자의 활동 및 교수 전략, 학습자의 활동을 제시한 것입니다.
평가	평가는 해당 학습모듈의 학습 정도를 확인할 수 있는 평가 준거 및 평가 방법, 평가 결과의 피드백 방법을 제시한 것입니다.

학습 1	촬영 대본 확정하기
학습 2	제작 스태프 조직하기
학습 3	촬영 장비 계획하기
학습 4	촬영 소품 계획하기

2-1. 기술 스태프 조직

학습 목표 • 프로그램 제작에 적합한 기술 스태프를 조직할 수 있다.

필요 지식 /

1. 기술 스태프의 구성
 프로그램의 장르에 따라 구성하는 기술 스태프는 많은 차이가 있다. 같은 장르의 프로그램이라도 그 형식이나 내용, 규모에 따라서 구성되는 기술 스태프의 종류와 인원 수는 천차만별이다.

1. 스튜디오 프로그램
 토크쇼, 종합 구성, 예능과 같은 스튜디오 프로그램은 부조정실과 스튜디오를 사용하여 제작하기 때문에 많은 기술 스태프가 필요하다.

학습은
 해당 NCS 능력단위요소 명칭을 사용하여 제시하였습니다. 하나의 학습은 일반교과의 '대단원'에 해당되며, 학습모듈을 구성하는 가장 큰 단위가 됩니다. 또한 하나의 직무를 수행하기 위한 가장 기본적인 단위로 사용할 수 있습니다.

학습 내용은
 NCS 능력단위요소별 수행준거를 기준으로 제시하였습니다. 일반교과의 '중단원'에 해당합니다.

학습 목표는
 학습 내용을 이수할 때 학습자가 갖춰야 할 행동 수준을 의미합니다. 따라서 수업시간의 과목 목표로 활용할 수 있습니다.

필요 지식은
 해당 NCS의 지식을 토대로 학습에 대한 이해와 성과를 제고하기 위해 반드시 알아야 할 주요 지식을 제시하였습니다. 필요 지식은 수행에 꼭 필요한 핵심 내용을 위주로 제시하여 교수자의 역할이 매우 중요하며, 이후 수행 순서와 연계하여 교수·학습으로 진행할 수 있습니다.

수행 내용 / 기술 스태프 구성표 작성하기

재료·자료

- 방송프로그램 제작 기획서 및 방송 대본, 콘티(continuity), 제작 일정, 운용표
- 장비 및 시설, 제작 시설 배정 의뢰서 및 배정표, 방송 기술 스태프 데이터베이스(DB) 자료

기기(장비·공구)

- 컴퓨터 등

안전·유의 사항

- 프로그램의 내용과 제작 방법을 분석하고, 각 스태프들의 역할을 신중하게 검토한다.

수행 순서

- 1 방송 대본이나 콘티(continuity), 큐 시트를 분석하고, 프로그램의 내용적 특성, 제작 과정에 대한 자료를 수집한다.
 - 2 프로그램 제작 방법을 결정한다.
1. 스튜디오 녹화를 할 것인가, 야외 촬영을 할 것인가 검토한다.

수행 tip

- 스태프의 결정은 스태프 간의 호흡을 중요시하여 선정해야 프로그램의 질을 향상시킬 수 있다.

수행 내용은

해당 학습모듈에서 제시한 내용 중 기술(skill)을 습득하기 위한 실습과제로 활용할 수 있습니다.

재료·자료는

수행 내용을 수행하는데 필요한 재료 및 준비물로 실습 시 활용할 수 있습니다.

기기(장비·공구)는

수행 내용에 필요한 기본적인 장비 및 도구를 제시하였습니다. 제시된 기기 외에도 수행에 필요한 다양한 도구나 장비를 활용할 수 있습니다.

안전·유의사항은

수행 내용을 수행하는 데 있어 안전상 주의해야 할 점 및 유의사항을 제시하였습니다. 실습 시 유념해야 하며, NCS의 고려사항도 추가적으로 활용할 수 있습니다.

수행 순서는

실습 과제의 진행 순서로 활용할 수 있습니다.

수행 tip은

수행 내용에서 실습을 용이하게 할 수 있는 아이디어를 제시하였습니다. 수행 tip은 지도상의 안전 및 유의사항 외에 전반적으로 적용되는 주안점 및 수행 과제 목적에 대한 보충설명, 추가사항 등으로 활용할 수 있습니다.

학습2 교수·학습 방법

교수 방법

- 방송 프로그램의 기술적 요소, 미술 구성 요소, 특수 촬영에 대해 설명한다.
- 방송 프로그램 제작에서 각 기술 스태프의 역할에 대해 설명한다.
- 방송 프로그램을 분석하고 필요한 기술 스태프를 구성할 수 있도록 지시한다.

학습 방법

- 방송 프로그램의 기술적 요소, 미술 구성 요소, 특수 촬영에 대해서 알아본다.
- 프로그램 제작에 필요한 기술 스태프의 역할을 이해하고, 기술 스태프 구성표를 작성한다.

교수·학습 방법은

학습 목표를 성취하는 데 필요한 교수 방법과 학습 방법을 제시하였습니다.

교수 방법은

해당 학습 활동에 필요한 학습 내용, 학습 내용과 관련된 자료명, 자료 형태, 수행 내용의 진행 방식 등에 대하여 제시하였습니다. 또한 학습자의 수업참여도 제고 방법 및 수업 진행상 유의사항 등도 제시하였습니다. 선수학습이 필요한 학습을 학습자가 숙지하였는지 교수자가 확인하는 과정으로 활용할 수도 있습니다.

학습 방법은

해당 학습 활동에 필요한 학습자의 자기주도 학습 방법을 제시하였습니다. 또한 학습자가 숙달해야 할 실기 능력과 학습 과정에서 주의해야 할 사항 등도 제시하였습니다. 학습자가 학습을 이수하기 전 반드시 숙지해야 할 기본 지식을 학습하였는지 스스로 확인하는 과정에 활용할 수 있습니다.

학습2

평 가

평가 준거

- 평가자는 학습자가 학습 목표를 성공적으로 달성하였는지를 평가해야 한다.
- 평가자는 다음 사항을 평가해야 한다.

학습 내용	학습 목표	성취수준 상 중 하
기술 스태프 조직	- 프로그램 제작에 적합한 기술 스태프를 조직할 수 있다.	
미술 스태프 조직	- 프로그램 제작에 적합한 미술 스태프를 조직할 수 있다.	
전문 스태프 조직	- 프로그램 특수 촬영을 위한 전문 스태프를 조직할 수 있다.	

평가 방법

- 사례 연구

학습 내용	평가 항목	성취수준 상 중 하
기술 스태프 조직	- 프로그램에서 기술적 요소의 파악 여부 - 기술 스태프의 역할 파악 여부 - 프로그램에 필요한 기술 스태프 구성표 작성 능력	

피드백

- 사례 연구
 - 프로그램을 선택하여 기술 스태프, 미술 스태프, 전문 스태프 구성표를 예시와 같이 작성하였는지 개인별 능력을 평가한 후, 그 결과를 모든 학습자에게 공유하도록 한다.

평가는

NCS 능력단위의 평가 방법과 평가 시 고려사항을 준용하여 작성합니다. 교수자와 학습자가 평가 항목별 성취수준 확인 시 활용할 수 있습니다.

평가 준거는

학습자가 학습을 어느 정도 성취하였는지 평가하기 위한 기준을 제시하고 있습니다. 학습 목표와 연계하여 단위수업 시간에 평가 항목 별 성취수준을 평가하는 데 활용할 수 있습니다.

평가 방법은

NCS 능력단위의 평가 방법을 참고하였으며, 평가 준거에 따른 평가 방법을 2개 이상 제시합니다. 평가 방법의 종류는 포트폴리오, 문제해결 시나리오, 서술형 시험, 논술형 시험, 사례 연구, 평가자 체크리스트, 작업장 평가 등이 있으며, NCS 능력단위 요소 별 수행 수준을 평가하는 데 가장 적절한 방법을 선정하여 활용할 수 있습니다.

피드백은

평가 후에 학습자들에게 평가 결과를 피드백하여 학습 목표를 달성하는 데 활용할 수 있습니다.

4. 참고 자료

참 고 자 료

- 교육부(2013). 섭외 및 제작스태프 구성(LM0803020104_13v1). 한국직업능력개발원.

참고 자료는

해당 학습모듈에 제시된 인용 자료의 출처를 제시하였습니다. 교수·학습의 과정에서 참고로 활용할 수 있습니다.

5. 활용 서식/부록

활 용 서 식

스튜디오 기술 스태프 구성표

직종	이름	연락처	소속	특이사항	비고
기술감독					
조명감독					

활용 서식은

평가 서식, 실습 시트 등 교수·학습 시 활용할 수 있는 다양한 서식들로 구성하였습니다. 수행에서 평가에 이르기까지 필요한 서식을 해당 모듈의 특성에 맞춰 개발하거나 기존의 양식을 활용하여 제시하였습니다.

부 록

[디지털 텔레비전 방송프로그램 음량 등에 관한 기준]

제정 2014. 11. 29. 미래창조과학부 고시 제2014-87호

제1장 총칙

제1조(목적) 이 고시는 방송법 제70조의2제1항에 따라 방송사업자가 디지털 텔레비전 방송프로그램 및 방송광고의 음량을 일정하게 유지하기 위해 필요한 사항을 규정함을 목적으로 한다.

부록은

활용 서식 이외에 교수·학습 과정에서 참고할 수 있는 자료가 있는 경우 제시하였습니다.

[NCS-학습모듈의 위치]

대분류	정보통신	
중분류	정보기술	
소분류	정보기술개발	

세분류		
SW아키텍처		
	능력단위	학습모듈명
응용SW엔지니어링	요구사항 확인	요구사항 확인
임베디드SW엔지니어링	데이터 입출력 구현	데이터 입출력 구현
DB엔지니어링	통합 구현	통합 구현
NW엔지니어링	정보시스템 이행	정보시스템 이행
보안엔지니어링	제품소프트웨어 패키징	제품소프트웨어 패키징
UI/UX엔지니어링	서버프로그램 구현	서버프로그램 구현
시스템SW엔지니어링	인터페이스 구현	인터페이스 구현
빅데이터플랫폼 구축	애플리케이션 배포	애플리케이션 배포
핀테크엔지니어링	애플리케이션 리팩토링	애플리케이션 리팩토링
데이터아키텍처	인터페이스 설계	인터페이스 설계
IoT시스템연동	애플리케이션 요구사항 분석	애플리케이션 요구사항 분석
인프라스트럭처 아키텍처 구축	기능 모델링	기능 모델링
	애플리케이션 설계	애플리케이션 설계

정적모델 설계	정적모델 설계
동적모델 설계	동적모델 설계
화면 설계	화면 설계
화면 구현	화면 구현
애플리케이션 테스트 관리	애플리케이션 테스트 관리
애플리케이션 테스트 수행	애플리케이션 테스트 수행
소프트웨어공학 활용	소프트웨어공학 활용
소프트웨어개발 방법론 활용	소프트웨어개발 방법론 활용
프로그래밍 언어 응용	프로그래밍 언어 응용
프로그래밍 언어 활용	프로그래밍 언어 활용
응용SW 기초 기술 활용	응용SW 기초 기술 활용
개발자 환경 구축	개발자 환경 구축
개발 환경 운영 지원	개발 환경 운영 지원

차 례

학습מוד의 개요	1
학습 1. 언어 특성 활용하기	
1-1. 언어별 특성 파악	3
1-2. 애플리케이션 구현 및 최적화	15
• 교수 · 학습 방법	42
• 평가	43
학습 2. 라이브러리 활용하기	
2-1. 라이브러리 선정	45
2-2. 라이브러리 구성 및 적용	57
• 교수 · 학습 방법	73
• 평가	74
참고 자료	76

프로그래밍 언어 응용 학습מוד의 개요

학습מוד의 목표

응용소프트웨어 개발에 사용되는 프로그래밍 언어의 특징과 라이브러리를 활용하여 기본 응용소프트웨어를 구현할 수 있다.

선수학습

개발자 환경 구축(2001020233_19v4), 프로그래밍 언어 활용(2001020231_19v4), 응용SW 기초 기술 활용(2001020232_19v4), 애플리케이션 테스트 수행(2001020227_19v5)

학습מוד의 내용체계

학습	학습 내용	NCS 능력단위 요소	
		코드번호	요소 명칭
1. 언어 특성 활용하기	1-1. 언어별 특성 파악	2001020230_19v4.1	언어 특성 활용하기
	1-2. 애플리케이션 구현 및 최적화		
2. 라이브러리 활용하기	2-1. 라이브러리 선정	2001020230_19v4.2	라이브러리 활용하기
	2-2. 라이브러리 구성 및 적용		

핵심 용어

저급/고급언어, 원시 코드(소스코드), 목적 코드(실행 파일), 리뷰, 리팩토링, 모듈, 패키지, 라이브러리

1-1. 언어별 특성 파악

학습 목표

- 프로그래밍 언어별 특성을 파악하고 설명할 수 있다.

필요 지식 /

① 프로그래밍 언어의 발전 과정과 유형 분류

초기 프로그래밍 언어는 부호였으나 컴퓨터 시스템의 발전과 함께 계속해서 발전해 왔다. 시대에 따른 발전 과정과 분류 체계는 다음과 같다.

1. 프로그래밍 언어의 발전 과정

프로그래밍 언어는 컴퓨터의 발전과 함께 계속하여 발전하였으며 어셈블리어에서 고급언어의 형태로 지속적으로 발전하고 있다.

1970년대 이전	1970년대	1980년대	1990년대	2000년대 이후
FORTRAN (1954)	PASCAL (1970)	ADA (1983)	Python (1991)	C# (2001)
CORBOL (1959)	C (1972)	C++ (1983)	HTML (1991)	Scala (2003)
BASIC (1964)	SMALLTALK (1972)	Objective-C (1986)	JAVA (1995)	Go (2009)
BCPL (1967)	SQL (1978)	Perl (1987)	PHP (1995)	Swift (2014)

출처: 집필진 제작(2021)

[그림 1-1] 프로그래밍 언어의 발전 과정

2. 프로그래밍 언어의 유형 분류

프로그래밍 언어별로 각각의 특성을 보유하고 있으며, 관점에 따라 프로그래밍 언어를 유형별로 분류할 수 있다.

(1) 개발 편의성 측면에 따른 분류

(가) 저급언어(Low-Level Language): 컴퓨터가 직접 이해할 수 있는 언어로 실행속도는 빠르나 기계마다 기계어가 상이하여 호환성이 없고 유지관리가 어렵다.

(나) 고급언어(High-Level Language): 개발자가 이해할 수 있도록 소스코드를 작성할 수 있는 언어로, 실행을 위해서는 번역 과정이 필요하다.

(2) 실행 및 구현 방식에 따른 분류

(가) 명령형 언어(Imperative Language): 컴퓨터가 동작해야 할 알고리즘을 통해 프로그래밍의 상태를 변경시키는 구문에 중점을 둔 방식으로 FORTRAN, C 등이 속한다.

(나) 함수형 언어(Functional Language): 함수의 응용을 강조하면서 자료의 처리는 수학적 함수의 연산으로 취급하고, 상태와 가변 데이터는 멀리하는 방식으로 LISP, Scala 등이 속한다.

(다) 논리형 언어(Logic Language): 논리 문장을 이용하여 프로그램을 표현하고 조건이 만족되면 연관된 규칙이 실행되는 방식으로 PROLOG 등이 속한다.

(라) 객체지향언어(Object-Oriented Language): 객체 간의 메시지 통신을 이용하여 동작하는 방식으로 JAVA, C++ 등이 속한다.

(3) 빌드(Build) 방식에 따른 분류

프로그램의 소스코드가 실행 가능한 형태로 변하는 과정을 빌드(Build)라고 하며 빌드 방식에 따라 분류할 수 있다.

(가) 컴파일 언어(Compile Language): 소스코드가 기계어 실행 파일로 빌드되는 방식이다. C, C++ 등이 속하며 실행속도가 높은 특징이 있다.

(나) 인터프리터 언어(Interpreter Language): 소스코드를 한 줄씩 번역하여 실행하는 방식이다. Python 등이 속하고 실시간 실행 및 분석이 가능한 특징이 있다.

(다) 바이트 코드 언어(Byte Code Language): 컴파일을 통해 가상머신이 번역할 수 있는 Byte Code로 변환되며, 가상머신은 다시 Native OS가 이해할 수 있는 기계어로 번역하는 방식이다. JAVA, Scala 등이 속한다.

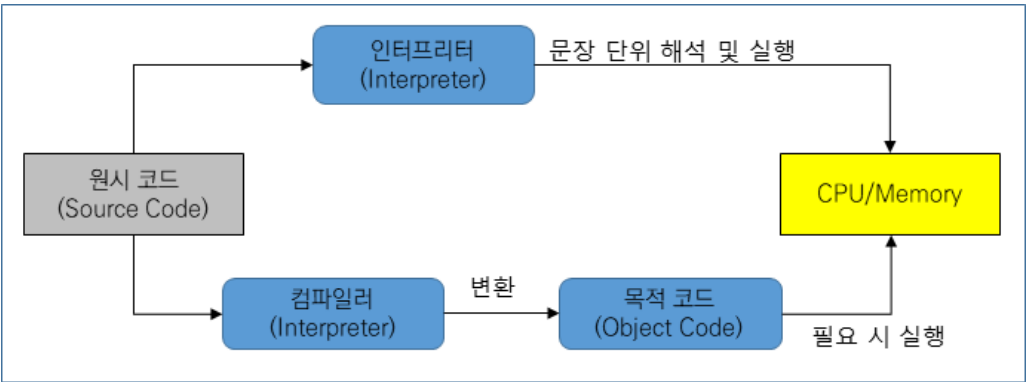
② 컴파일러(Compiler)와 인터프리터(Interpreter)

고급언어로 작성한 소스코드의 경우 컴퓨터가 이해할 수 없으므로 컴파일러나 인터프리터를 이용하여 컴퓨터가 이해하고 실행할 수 있는 기계어 코드로 번역을 수행한다.

1. 컴파일러와 인터프리터 동작 방식

컴파일러의 경우 소스코드를 목적 코드로 변환하여 실행하는 방식이며, 인터프리터

의 경우 문장 단위로 읽어들이어 해석을 하여 실행한다.



출처: 집필진 제작(2021)
[그림 1-2] 컴파일러와 인터프리터 동작 방식

2. 컴파일러와 인터프리터 비교

컴파일 방식의 경우 실행속도가 빠르고 보안적인 측면에서는 유리한 장점이 있으나 매번 빌드 작업을 거쳐야 하는 불편함이 존재하며, 인터프리터 방식의 경우 즉시 수정 및 실행이 가능한 장점은 있으나 처리속도가 느린 단점이 있다.

<표 1-1> 컴파일러 및 인터프리터 비교

구분	컴파일러	인터프리터
개발 편의성	코드를 수정하고 실행이 필요한 경우 재컴파일 필요	코드 수정 후 즉시 실행 가능
번역 단위	전체 소스코드	문장 단위
실행 파일 및 속도	실행 파일 생성 처리속도 빠름	실행 파일 미생성 처리속도 느림
메모리 할당	실행 파일 생성 시 사용	할당하지 않음
오류 확인 및 처리	전체 코드에 대한 컴파일 수행 시 발생한 오류 확인 가능	프로그램 실행 후 오류가 발생한 문장 이후의 코드는 실행하지 않음
파일 용량 및 보안	실행 파일 전체를 처리해야 하므로 용량이 크며, 원시 코드의 유출 가능성이 상대적으로 낮음	원시 코드만 처리하면 되므로 용량이 상대적으로 작고, 원시 코드의 유출 가능성이 높음
주요 언어	C, C++, JAVA	Python, Javascript, Ruby

수행 내용 / 언어별 특성 파악하기

재료 · 자료

- 요구사항 관련 문서
- 개발 표준 관련 문서
- 프로그래밍 언어의 발전 과정에 대한 자료
- 프로그래밍 언어별 작성 예시 코드

기기(장비 · 공구)

- 전산 장비: 인터넷, 컴퓨터, 프린터, 복사기, 빔 프로젝터
- 지원 도구: 문서 작성 도구, 문서 발표 도구, 소프트웨어 개발 도구

안전 · 유의 사항

- 프로그래밍 언어의 시대별 발전 과정에 대해 사전에 확인한다.
- 프로그래밍 언어의 종류에 대해 사전에 확인한다.
- 프로그래밍 언어 특성의 이해를 위해 언어별 작성 코드 예시를 사전에 준비한다.

수행 순서

① 프로그래밍 언어의 발전 과정과 개별 특성을 파악한다.

1. 프로그래밍 언어의 발전 과정과 영향 관계를 파악한다.

(1) 시기별 프로그래밍 언어 발전 과정

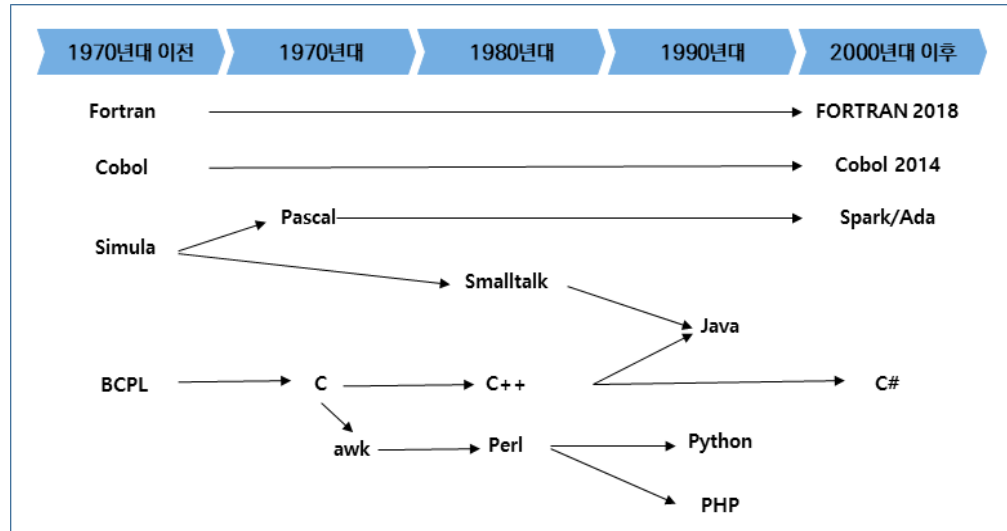
컴퓨터의 발전과 시대의 흐름에 따라 발전하였으며, 시기에 따른 특징을 가지고 있으므로 시기별로 등장한 프로그램 언어를 파악한다.

〈표 1-2〉 시기별 프로그래밍 언어 발전 과정

시기	주요 프로그래밍 언어
1960년 이전	ASSEMBLY, FORTRAN, LISP
1960년대	COBOL, PL/I, BASIC
1970년대	PASCAL, C, SMALLTALK, PROLOG
1980년대	ADA, C++, PERL, PYTHON
1990년대	RUBY, JAVA, JAVASCRIPT, PHP, VISUAL BASIC
2000년 이후	C#, SCALA, GO, CLOJURE, CEYLON, KOTLIN, DART 등

(2) 프로그래밍 언어 간 관계

초기 등장 후 꾸준히 활용되고 있는 언어와 기존 언어에서 영향을 받아 등장한 프로그래밍 언어에 대해 파악한다.



출처: 집필진 제작(2021)

[그림 1-3] 프로그래밍 언어의 영향 관계

② 프로그래밍 언어의 분류 체계와 언어의 종류를 파악한다.

1. 프로그래밍 언어의 분류 체계를 파악한다.

프로그래밍 언어의 개발 편의성, 실행 및 구현 방식, 빌드 방식에 따른 분류 체계와 특징에 대해 파악한다.

(1) 개발 편의성 측면에 따른 분류 체계를 파악한다.

언어 사용 주체에 따라 컴퓨터가 이해하기 쉽게 작성된 저급언어와 사람이 이해하기 쉽게 작성한 고급언어로 분류될 수 있으며, 분류에 따른 특성을 파악한다.

〈표 1-3〉 개발 편의성에 따른 분류 체계

구분	설명
저급언어	기계가 이해하기 쉽게 작성된 언어. 실행속도가 빠른 장점이 있으나 추상화 수준이 낮고 배우거나 유지관리가 힘든 단점 등으로 현재는 거의 사용하지 않는다.
고급언어	사람이 이해하기 쉽게 작성된 언어. 가독성이 높고 번역 과정(컴파일러, 인터프리터)이 필요하며 대부분의 프로그래밍에서 사용한다.

(2) 실행 및 구현 방식에 따른 분류 체계를 파악한다.

실행 및 구현 방식에 따라 명령형, 함수형, 논리형, 객체지향형으로 구분할 수 있으며 각 분류 기준에 따른 특징을 파악한다.

〈표 1-4〉 실행 및 구현 방식에 따른 분류 체계

구분	설명
명령형 언어	컴퓨터에 저장된 명령어들이 순차적으로 실행되는 프로그래밍 방식으로 절차형 언어라고도 한다.
함수형 언어	명령형 언어가 상태 변경을 강조하는 반면 함수형 언어는 함수의 응용을 강조하여 수학적인 수식 등의 함수들로 프로그램을 구성하여 호출하는 방식의 언어이다.
논리형 언어	논리적인 문장 구조를 이용하여 프로그램을 표현하고 계산을 수행하는 구조로 추론과 관계 규칙에 의해 원하는 결과를 얻는다.
객체지향형 언어	객체 간의 관계에 초점을 두고 기능을 중심으로 메소드를 구현하는 방법으로 상속, 캡슐화, 다형성, 추상화 등의 특징을 가지고 있다.

(3) 빌드 방식에 따른 분류 체계를 파악한다.

빌드 방식에 따라 컴파일 언어, 인터프리터 언어, 바이트 코드 언어로 구분할 수 있으며 각 분류 기준에 따른 특징을 파악한다.

〈표 1-5〉 빌드 방식에 따른 분류 체계

구분	설명
컴파일 언어	소스코드를 컴파일러를 통해 실행 가능한 형태의 기계어로 미리 번역하는 과정이 필요하며, 실행에 필요한 정보가 미리 계산되어 구동하는 시간은 오래 걸리지만 실행속도가 빠르다.
인터프리터 언어	별도의 컴파일 과정 없이 바로 실행 가능한 장점이 있으나 소스코드를 하나씩 번역하여 실행함으로써 실행속도는 느린 특징이 있다.
바이트 코드 언어	컴파일을 통해 고급언어를 중간 언어로 변환한 후 가상머신에 의해 번역을 실행하는 방식으로 여러 환경에서 사용 가능한 특징이 있다.

2. 분류 체계별 프로그래밍 언어의 종류를 파악한다.

각 분류 체계에 포함되는 언어의 종류 및 각 언어가 가지고 있는 특징을 파악한다.

(1) 개발 편의성에 따른 프로그래밍 언어의 종류를 파악한다.

개발 편의성에 따라 저급언어와 고급언어로 분류되는 언어의 종류를 파악한다.

〈표 1-6〉 개발 편의성에 따른 프로그래밍 언어 종류

구분	주요 언어
저급언어	기계어, 어셈블리어
고급언어	C, C++, JAVA, PYTHON, .NET 등 대다수 언어

(2) 실행 및 구현 방식에 따른 프로그래밍 언어의 종류를 파악한다.

실행 및 구현 방식에 따라 명령형 및 함수형, 논리형, 객체지향형 언어로 분류되는 언어의 종류를 파악한다.

〈표 1-7〉 실행 및 구현 방식에 프로그래밍 언어 종류

구분	주요 언어
명령형 언어	FORTTRAN, COBOL, PASCAL, C
함수형 언어	SCHEME, ERLANG, LISP, ML
논리형 언어	PROLOG
객체지향형 언어	JAVA, C++, SMALLTALK, PYTHON

(3) 빌드 방식에 따른 분류 체계를 파악한다.

빌드 방식에 따라 컴파일, 인터프리터, 바이트 코드 언어로 분류되는 언어의 종류를 파악한다.

〈표 1-8〉 빌드 방식에 따른 프로그래밍 언어 종류

구분	주요 언어
컴파일 언어	FORTTRAN, PASCAL, C, C++
인터프리터 언어	별도의 BASIC, PROLOG, LISP, SNOBOL
바이트 코드 언어	JAVA, SCALA

③ 프로그래밍 언어의 개별 특성과 사용 동향을 파악한다.

개별 프로그래밍 언어의 특성을 파악하고 각 언어의 예시 코드 및 사용 현황과 추이에 대해 파악한다.

1. 프로그래밍 언어의 개별 특성을 파악한다.

프로그래밍 언어의 분류 체계를 참고하여 각 개별 언어에 대한 특성을 파악한다.

〈표 1-9〉 프로그램 언어별 설명 및 특성

프로그램 언어	설명 및 특성
FORTTRAN	IBM에서 과학적인 계산을 하기 위해 시작된 언어로 간결하고 엄격한 구문 형식의 언어. 컴퓨터 시스템에 대해 많은 관련 지식이 필요하며 기상예측, 자원탐사, 우주 항공, 유체 및 구조해석 등의 과학계산 전문 분야에 활용
COBOL	미국 국방성에 의해 개발되었으며 비즈니스, 금융, 회사/정부 관리 시스템에 주로 사용. 메인프레임 컴퓨터의 레거시 응용프로그램들에 사용되고 있으며, 비교적 프로그램 크기가 크고 구문이 복잡
PASCAL	잘 짜인 구조와 간결성으로 프로그래밍 교육을 위해서 널리 사용되었으며, 파스칼에 객체지향 개념을 포함한 오브젝트 파스칼(Object Pascal), 오브젝트 파스칼을 일부 변형한 델파이(Delphi) 프로그래밍 언어로 발전
C	UNIX 운영체제 구현에 사용되는 언어이며 효율적인 실행과 간결한 문법, 효과적인 포인터 타입 제공이라는 특징으로 인해 많이 사용되고 있는 시스템 프로그래밍 언어
C++ /C#	<ul style="list-style-type: none"> - C++: C 언어에서 발전한 언어로 다중 상속 등의 기능을 제공하는 객체지향 프로그래밍 언어 - C#: C와 C++의 발전된 형태로 .NET 환경에 맞춰 설계되었으며, 사용자 인터페이스를 쉽게 만들 수 있는 컴포넌트 기능도 제공
PHP	웹 개발에 특화된 언어로 다양한 프레임워크를 지원하고 특별한 컴포넌트 설치하지 않아도 다양한 처리가 가능
JAVA	C++에 비해 단순하고 분산환경과 객체지향, 보안성을 지원하고 컴파일을 통해 class 파일을 생성하며 가상머신(JVM)에서 실행
JAVASCRIPT	HTML, CSS와 함께 웹을 구성하는 핵심 요소로 거의 모든 웹 브라우저에 스크립트 엔진(인터프리터)이 존재. 웹 페이지의 동작 구현이 가능하고 빠른 개발과 확장성이 뛰어나지만 다른 언어에 비해 상대적으로 보안이나 성능이 부족
PYTHON	배우기 쉽고 이식성이 좋은 언어로 다양한 함수들도 많이 제공되어 최근 트렌드와 맞물려 스타트업과 글로벌 기업 등에서도 많이 사용하는 언어
GOLANG	Google에서 만든 언어로 짧게는 GO라고도 부르며 내장 라이브러리가 많이 지원되고 C언어의 문법과 유사하나 제어 구조를 가지고 빠른 컴파일이 가능함
KOTLIN	JAVA보다 간결한 문법을 가지고 있는 JVM 기반의 언어로 JAVA와의 상호 운용이 100% 지원되고 2019년 구글이 안드로이드의 공식 언어로 추가
R	통계 및 그래프 작업을 위한 인터프리터 프로그래밍 언어로 수많은 통계 관련 패키지가 개발되어 있고 빅데이터 분석 및 기계학습 등에 유용

2. 프로그래밍 언어의 예시 코드를 파악한다.

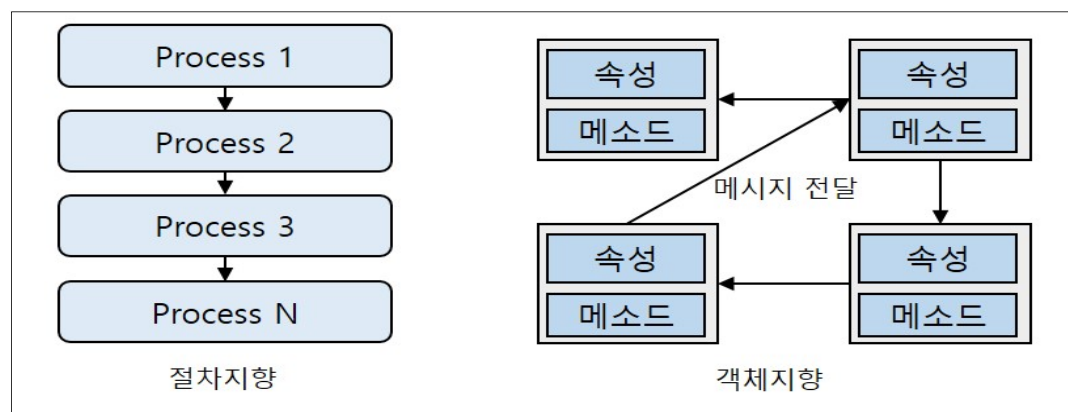
프로그래밍 언어의 개별적인 특성 이외에 간단한 예시 코드를 확인하여 구현 관점에서의 특징을 파악한다.

〈표 1-10〉 프로그램 언어별 문자 출력 예시 코드

프로그램 언어	문자열 출력 소스코드 예시
FORTTRAN	Print *, "Hello, NCS!!"
C	<pre>#include <stdio.h> int main(void){ puts("Hello, NCS!!"); return 0; }</pre>
JAVA	<pre>public static void main(String[] args) { System.out.println("Hello, NCS!!"); }</pre>
KOTLIN	<pre>fun main(args:Array<String>){ println("Hello, NCS!!") }</pre>
PHP	<pre><?php echo ("Hello, NCS!!"); ?></pre>
JSP	<pre><% out.println ("Hello, NCS!!"); %></pre>
PYTHON	print("Hello, NCS!!")
JAVASCRIPT	<pre><script>alert("Hello, NCS!!");</script> <script>document.write("Hello, NCS!!");</script></pre>

3. 절차형 언어와 객체지향 언어의 차이를 파악한다.

절차지향의 경우 절차(실행 순서)가 중점이 되고, 객체지향의 경우 객체의 종류와 속성 등에 더 중점을 둔 개발 패러다임을 의미한다.



출처: 집필진 제작(2021)

[그림 1-4] 절차형 언어와 객체지향 언어의 개념 비교

(1) 객체지향의 구성요소와 특징에 대해 파악한다.

객체지향의 경우 클래스와 객체, 메소드로 구성되며 상속과 추상화, 다형성, 캡슐화와 같은 특징을 가진다.

(가) 객체지향의 구성요소를 파악한다.

- 1) 객체(Object): 개체와 속성, 메소드로 구성된 클래스의 Instance를 의미한다.
- 2) 클래스(Class): 공통된 특성(속성, 연산)을 가지는 객체 집합으로 객체 타입을 정의하고 생성하는 틀이다.
- 3) 메시지(Message): 객체 간의 상호작용은 메시지를 통해 이루어지며 메시지는 객체에서 객체로 전달된다.

(나) 객체지향의 특징을 파악한다.

- 1) 캡슐화(Encapsulation): 연관된 데이터와 데이터를 처리하는 함수를 함께 묶어 외부에는 필요한 인터페이스만을 노출한다.
- 2) 정보 은닉(Information Hiding): 다른 객체에게 자신의 필드 및 메소드 등을 은닉하고 자신의 연산만을 통하여 접근을 허용한다.
- 3) 추상화(Abstraction): 불필요한 부분은 생략하고 주어진 문제나 시스템 중에서 중요한 부분에 집중하여 모델링한다.
- 4) 상속(Inheritance): 하위 클래스에서 상위 클래스의 속성과 메소드를 물려받는 기법으로 클래스와 객체의 재사용이 가능하다.
- 5) 다형성(Polymorphism): 하나의 메시지에 대해 각 객체의 고유한 방법으로 응답한다.

(2) 절차형 언어와 객체지향 언어의 차이를 파악한다.

절차형 언어와 객체지향 언어의 구성, 구현 방식, 관련 언어, 장점과 단점 등에 대해 파악한다.

〈표 1-11〉 절차지향과 객체지향 차이

구분	절차지향	객체지향
구성	함수	객체
구현 방식	전체적인 기능 동작을 고려한 각 단계별 기능 구현	필요한 속성의 객체 모델링 후 상호작용 기능 구현
접근제어	없음	가능(public, private)
상속/다형성	없음	가능
보안성	낮음	높음
장점	프로그램 흐름을 쉽게 추적 가능 복잡도가 단순하고 실행속도가 상대적으로 빠름	뛰어난 재사용 및 확장성, 유지보수의 용이성 보유 규모가 크고 협업이 필요한 대형 프로젝트 적합
단점	큰 프로젝트의 경우 구조 복잡, 중복 코드 발생, 유지성이 높아 신규 기능 추가 등이 어려움	상대적으로 속도가 느리고 메모리 사용률이 높음 설계 과정에 많은 시간 소요
언어	C, Fortran, Pascal	JAVA, C++, Python

4. 프로그래밍 언어별로 사용 용도를 확인한다.

웹 사이트, 모바일, 게임 개발 등에 따라 사용되는 언어가 완전히 상이할 수 있으므로 프로그래밍 언어별로 어떤 목적에 주로 사용되는지 확인한다. 또한, 파이썬이나 자바에서 실행속도가 중요한 부분은 상대적으로 실행속도가 빠른 C로 개발하여 상호 보완적인 역할을 할 수도 있다.

(1) 웹(프론트엔드): HTML, CSS, Javascript, JSP, PHP, ASP 등

(2) 모바일 개발

(가) 안드로이드: JAVA, Kotlin 등

(나) iOS: Object C, Swift 등

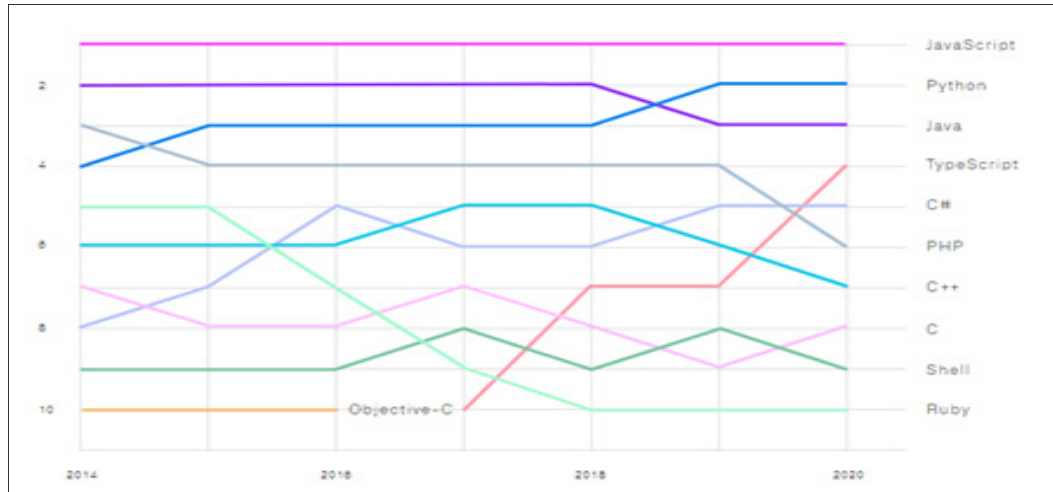
(3) 데스크톱: C, C++, C# 등

(4) 게임: C, C++, Unity 등

(5) AI/빅데이터 분석: Python, R 등

5. 프로그래밍 언어의 사용 현황과 추이에 대해 파악한다.

어떤 언어가 많이 활용되는지 여부와 각 언어의 사용 추이를 파악한다. [GitHub](#)의 통계 자료를 기준으로 2020년은 JavaScript, Python, Java가 많이 사용되었으며, Python의 경우 최근 트렌드인 인공지능, 빅데이터 분석 등에 활용됨에 따라 사용량이 증가한 것을 확인할 수 있다.



출처: The 2020 State of The OCTOVERSE(<https://octoverse.github.com/>). 2021. 06. 10. 스크린샷.
[그림 1-5] GitHub에서 많이 사용되고 있는 프로그래밍 언어

깃허브의 다양한 통계 정보 제공 사이트(<https://octoverse.github.com/>)

세계 최대 오픈소스 저장소를 운영하는 GitHub에서 사용 지역(Asia, Europe 등) 및 많이 사용되고 있는 언어 등의 다양한 통계 정보를 제공하는 사이트.

사용되고 있는 언어의 순위가 언어 자체의 좋고 나쁨을 의미하는 것은 아니며, 개발 트렌드를 파악하기 위한 용도로 활용하는 것이 적절하다.

수행 tip

- 일반적으로 많이 사용되고 있는 언어의 특징은 다양한 환경에서 사용할 수 있으며 확장성이 좋은 언어이다.
- 언어 사용 통계 등을 통하여 IT산업에서 많이 활용되는 언어의 확인이 가능하다.
- 각 프로그램 언어의 예시 코드를 확인하면 특징을 이해하는 데 도움이 된다.

1-2. 애플리케이션 구현 및 최적화

학습 목표

- 파악된 프로그래밍 언어의 특성을 적용하여 애플리케이션을 구현할 수 있다.
- 애플리케이션을 최적화하기 위해 프로그래밍 언어의 특성을 활용할 수 있다.

필요 지식 /

① 코드 인스펙션(Code Inspection)

프로그램 소스코드를 실행하지 않고 코드상에서의 잠재적인 오류와 표준 미준수 등의 결함을 사전에 식별하여 개선하는 공식적인 리뷰(Review) 기법

1. 인스펙션 수행 절차

중재자(Moderator), 저자(Author), 진행자(Reader), 검토자(Inspector), 기록자(Recorder) 등의 역할 담당자를 지정하여 진행한다. 인스펙션은 개발 초기 단계에서 결함을 식별하여 수정함으로써 오류로 인한 비용 최소화, 코드 품질 향상, 유사 결함의 발생 예방 및 유지보수 효율성 향상을 목적으로 진행한다.

〈표 1-12〉 인스펙션 수행 절차 및 수행 내용

단계	수행 내용
계획(Planning)	인스펙션 대상 산출물 선정 및 대상자를 구성. 인스펙션 대상 산출물을 사전에 배포하고 날짜와 시간 및 장소를 공지한다.
개관(Overview)	참여자를 대상으로 산출물에 대한 이해도를 높여 인스펙션의 효과성을 향상시킬 수 있으며 생략 가능하다.
준비(Preparation)	구성원이 개별적으로 산출물에 대해 숙지하고 체크리스트를 활용하여 결함 부분에 대해서 기록한다.
검토회의(Meeting)	산출물을 함께 검토하며 준비단계나 검토회의에서 식별된 결함 부분에 대해 집중한다.
재작업(Rework)	검토회의에서 발견된 결함에 대해 수정한다.
추적(Follow-up)	결함이 정상적으로 수정되었는지 최종 확인하고 인스펙션을 종료한다.

2. 코드리뷰 기법 비교

프로젝트 상황 및 코드의 중요도에 따라 동료검토, 워크스루, 테크니컬리뷰, 인스펙션 등의 기법을 선택하여 적용할 수 있다.

〈표 1-13〉 코드리뷰 기법 비교

단계	수행 내용
동료검토(Peer Review)	별도의 절차 없이 비공식으로 계획 없이 임의적으로 실시되며, 개발자가 동료와 코드 및 산출물의 결함을 식별하는 기법
워크스루(Walk Through)	개발자가 리뷰의 주제와 시간을 정해서 발표를 하고 동료들로부터 의견이나 아이디어를 듣는 시간을 가질 수 있으며, 사례에 대한 정보 공유나 아이디어 수집을 위해서 사용될 수 있다.
Inspection	역할과 절차, 체크리스트를 기준으로 결함을 식별하는 공식적인 리뷰 기법

② 리팩토링(Refactoring)

SW의 원래 기능은 유지하면서 소스코드의 내부 구조를 수정 및 보완하여 가독성, 성능 향상 및 로직을 개선하는 기법이다.

1. 주요 리팩토링 대상 및 방법

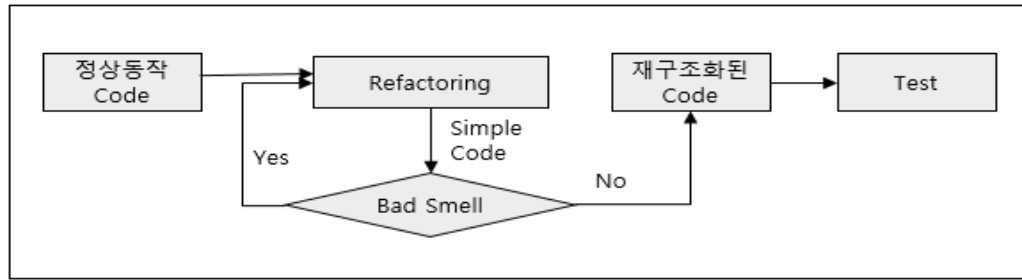
리팩토링 대상인 나쁜 냄새(Bad Smell)는 오류는 아니지만 문제를 유발할 수 있거나 설계 결함을 가지고 있는 부분으로 잠재적인 오류 예방과 성능 향상, 불필요한 코드 제거, 유지보수 비용 절감, 재사용성 향상 등을 위해 개선한다.

〈표 1-14〉 리팩토링 대상 및 적용 방법

리팩토링 대상	리팩토링 방법
중복된 코드(Duplicated Code)	중복을 제거해야 한다.
긴 메소드(Long Method)	메소드를 적정 수준의 크기로 나누어야 한다.
긴 파라미터 리스트(Long Parameter List)	파라미터 개수를 줄여야 한다.
게으른 클래스(Lazy Class)	자식 클래스와 부모 클래스의 차이가 없으면 합친다.
주석(Comment)	주석이 없어도 코드를 이해할 수 있도록 소스코드를 변경한다.
메시지 체인(Message Chain)	특정 객체를 얻기 위한 다수 객체는 간소화한다.
미들 맨(Middle Man)	다른 클래스로 위임하는 역할만 담당하는 클래스 제거를 검토한다.
불완전 라이브러리(Incomplete Library)	불완전 시 필요 부분 추가 구성한다.
스위치 명령문(Switch Statements)	지나치게 많은 case를 사용하는 Switch 문장은 코드 중복의 신호이다.

2. 리팩토링 수행 절차

정상 동작하는 코드를 대상으로 Bad Smell이 있는 부분을 지속적으로 개선하여 재구조화하며, Bad Smell은 코드 인스펙션과 PMD 등의 Tool을 이용하여 식별할 수 있다.



출처: 집필진 제작(2021)
[그림 1-6] 리팩토링 수행 절차

③ OWASP TOP 10

OWASP(The Open Web Application Security Project)는 웹 애플리케이션에 대한 보안 프로젝트이다. OWASP TOP 10은 웹 애플리케이션 취약점 중에서 빈도가 많이 발생하고 보안상 영향을 크게 줄 수 있는 10가지를 선정하여 발표한다.

〈표 1-15〉 OWASP TOP 10 항목 및 내용(2017년 발표 기준)

항목	상세 내용
Injection(인젝션)	웹 애플리케이션에 비정상적인 명령어나 Query 등을 보내 공격자가 시스템에 불법적으로 접근할 수 있는 취약점
Broken Authentication(취약한 인증)	잘못 구현된 인증이나 Session 관리 기능으로 인해 일시적 혹은 영구적으로 공격자가 다른 사용자의 권한을 획득할 수 있는 취약점
Sensitive Data Exposure(민감한 데이터 노출)	개인 식별 정보나 신용 정보 등의 민감 데이터 저장 및 전송 시 노출 취약점
XML External Entities(XXE) (XML 외부 개체)	XML 문서에서 External Entity를 이용하여 공격자가 의도하는 외부 URL을 실행하여 서버의 로컬파일 정보를 출력하거나 서비스 거부 공격을 수행할 수 있는 취약점
Broken Access control (취약한 접근 통제)	다른 사용자의 계정 접근, 중요 데이터 열람 및 수정, 액세스 권한 변경 등의 악의적인 행위가 가능한 취약점
Security misconfigurations (잘못된 보안 구성)	안전하지 않은 구성, 잘못된 구성으로 HTTP Header나 민감 정보가 포함된 에러 메시지 등으로 인한 취약점
Cross Site Scripting (XSS) (크로스 사이트 스크립팅)	웹 페이지에 악성 스크립트를 삽입할 수 있는 취약점으로 사용자의 정보(쿠키, 세션 등)를 탈취하거나 악의적인 사이트로 이동할 수 있는 취약점
Insecure Deserialization(안전하지 않은 역직렬화)	데이터를 역직렬화하는 과정에서 원격코드 실행이나 권한 상승 등이 가능한 취약점
Using Components with Known Vulnerabilities(알려진 취약점이 있는 구성요소 사용)	취약한 컴포넌트가 악용되는 경우 서버가 장악되거나 심각한 데이터 손실을 발생할 수 있는 취약점
Insufficient logging and monitoring (불충분한 로깅 및 모니터링)	충분하지 않은 로깅과 모니터링을 통해 시스템을 추가로 공격하고 데이터를 변조하거나 추출, 파괴 가능한 취약점

출처: OWASP Top Ten. <https://owasp.org/www-project-top-ten/>에서 2021. 08. 16. 검색.

④ 정적 분석과 PMD(Programming Mistake Detector)

1. 정적 분석(Static Analysis)

동적 분석(Dynamic Analysis)과 상대되는 개념으로 **소프트웨어를 실행하지 않고 코드 레벨에서 분석하는 방법**으로 소스코드의 실행 없이 코드의 의미를 분석해 결함을 찾아내는 코드 분석 기법. 오류가 존재하는 부분을 사전에 식별하여 영향 범위를 최소화할 수 있고, 테스트 단계에서의 부담을 줄여 프로그램의 품질을 높일 수 있다.

2. PMD(Programming Mistake Detector)

응용프로그램 코드의 문제를 사전에 확인할 수 있는 정적 분석을 위한 오픈 소스코드 분석 도구이며, 기본 제공 규칙 세트가 포함되어 있고 사용자 지정 규칙을 작성할 수 있다. 단독 형태로도 사용할 수 있고 이클립스 등과 같은 IDE에 플러그인 형태로 배포되어 누구나 사용할 수 있다.

〈표 1-16〉 PMD 기본 룰셋 및 내용 예시

기본 Rule	내용 설명
EmptyFinallyBlock	비어 있는 Finally 구문 단락
EmptyCatchBlock	비어 있는 Catch 구문 단락
UnnecessaryReturn	불필요한 Return 구문
DuplicateImports	Import문이 중복 선언
AvoidReassigningParameters	넘겨받는 메소드 Parameter 값을 직접 변경
EqualsNull	Null 값과 비교하기 위해 Equals 메소드를 사용
UnnecessaryParentheses	불필요한 괄호 사용
VariableNamingConventions	Final이 아닌 변수는 밑줄을 포함할 수 없음
StringToString	String 객체에서 toString() 함수를 사용
UnusedPrivateField	사용되지 않는 Private field
UnusedPrivateMethod	사용되지 않는 Private Method

재료 · 자료

- 요구사항 관련 문서
- 개발 표준 관련 문서
- 디자인 패턴 및 리팩토링 관련 자료
- 애플리케이션 보안 관련 자료

기기(장비 · 공구)

- 전산 장비: 인터넷, 컴퓨터, 프린터, 복사기, 빔 프로젝터
- 지원 도구: 문서 작성 도구, 문서 발표 도구, 소프트웨어 개발 도구

안전 · 유의 사항

- 사용자 환경(OS 및 버전 등)에 맞는 개발 언어 및 툴을 설치한다.
- 작성 코드의 버전이 관리될 수 있도록 버전 관리에 유의한다.
- 사용 언어에 따른 조건문과 반복문의 사용법을 파악하여 불필요한 연산이 되지 않도록 유의한다.
- 컴파일 및 실행 오류 등이 발생하는 경우 디버그 모드를 이용하여 오류를 해결한다.

수행 순서

① 프로그래밍 언어의 특성을 적용하여 애플리케이션을 구현한다.

애플리케이션 구현을 위해서는 프로그램을 개발할 수 있는 환경 구성이 필요하며, 프로그래밍 언어의 특성에 맞는 파일과 개발 도구를 설치한다.

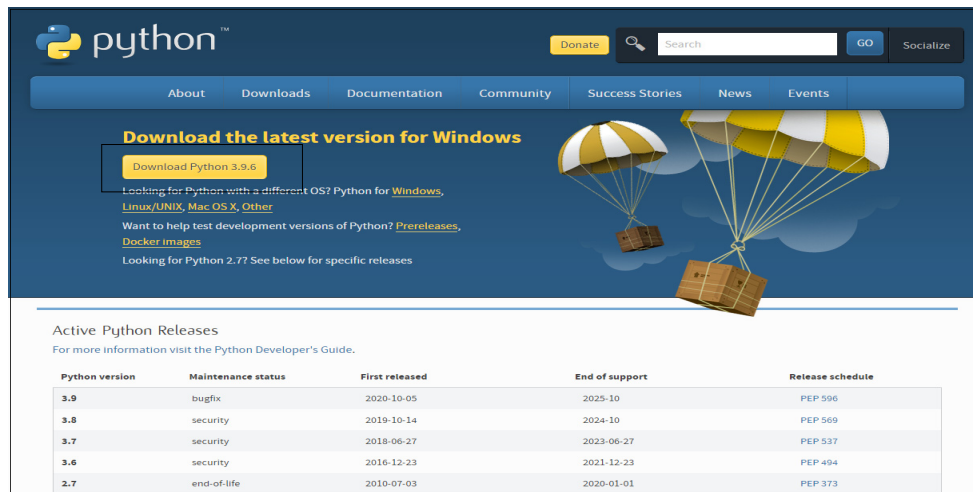
1. 프로그래밍 언어를 사용하기 위한 환경을 구성한다.

프로그래밍 언어의 설치 및 구현, 실행 등의 과정에서 언어 특성에 따른 차이점의 이해를 위하여 파이썬과 자바 개발환경을 구성한다. 자바와 파이썬 모두 메모장과 같은 텍스트 에디터와 CMD 등의 CLI(Command Line Interface)를 통해 개발 및 실행은 가능하나 효율적인 진행을 위해 IDE(Integrated Development Environment)를 설치한다.

(1) 파이썬 사용을 위한 개발환경을 구성한다.

(가) 파이썬 공식 사이트에서 설치 파일을 다운받는다.

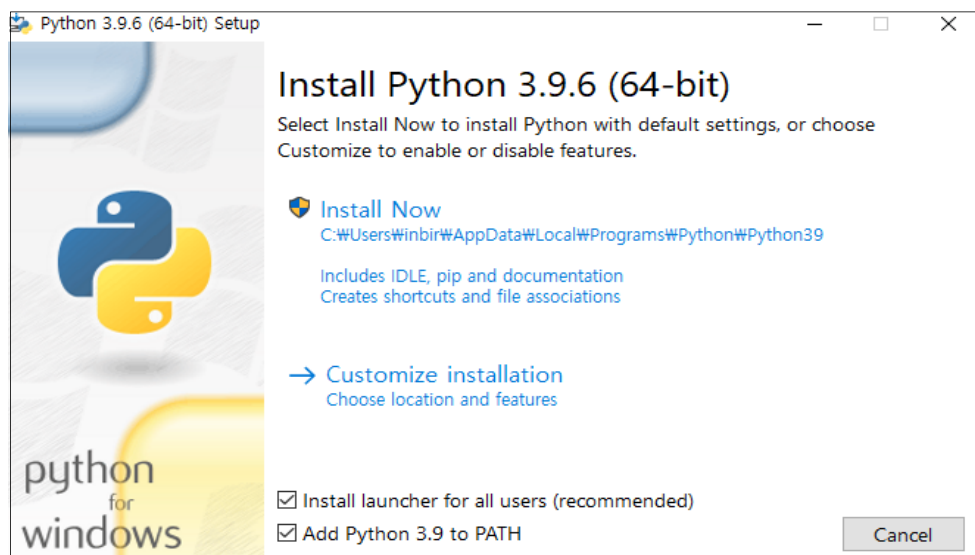
파이썬은 오픈소스 소프트웨어이며 공식 사이트(<https://www.python.org/>)에서 사용자 환경에 맞는 설치 파일을 다운받는다.



출처: 파이썬 공식 사이트(<https://www.python.org/downloads/>). 2021. 07. 10. 스크린샷.
[그림 1-7] 파이썬 설치 파일 다운로드

(나) 다운로드한 설치 파일을 PC에 설치한다.

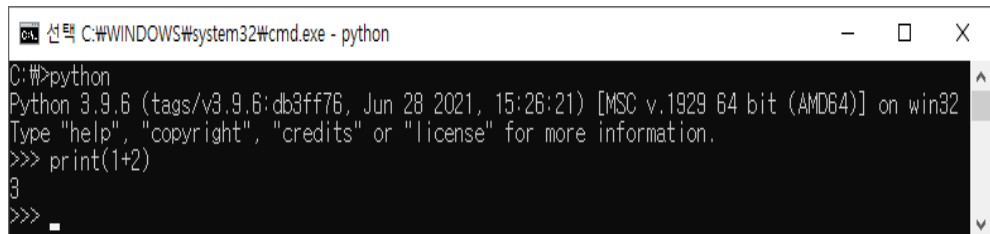
다운로드한 설치 파일을 실행하여 설치를 진행하고, 설치 위치는 사용자 환경에 맞춰 지정한다.



출처: 집필진 제작(2021)
[그림 1-8] 파이썬 설치 화면

(다) 정상적으로 설치되었는지 확인한다.

정상 설치 여부를 확인하기 위해 CMD 창을 띄워 python을 입력하고 간단한 연산 결과를 출력하는 명령문을 실행하여 연산 결과가 정상적으로 표시되는지 확인한다.

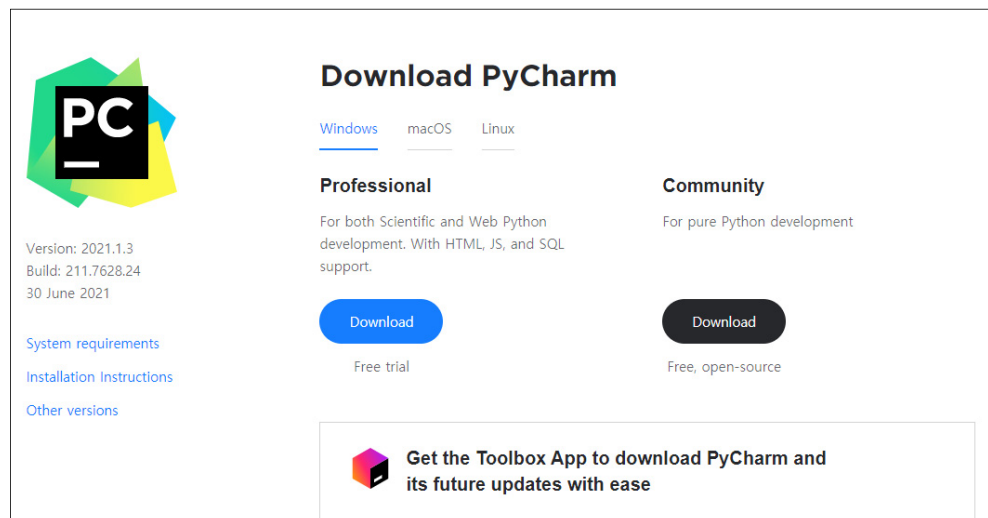


```
C:\>python
Python 3.9.6 (tags/v3.9.6:db3ff76, Jun 28 2021, 15:26:21) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print(1+2)
3
>>>
```

출처: 집필진 제작(2021)
[그림 1-9] 파이썬 설치 확인

(라) 파이썬 개발 툴을 다운로드한다.

파이썬 IDE인 PyCharm은 관련 공식 사이트에서 사용자 환경에 맞는 설치 파일을 다운받아 설치를 진행한다.

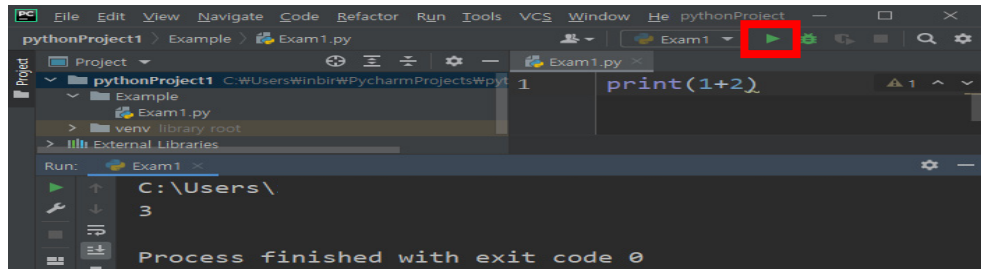


출처: PyCharm 다운로드 사이트(<https://www.jetbrains.com/pycharm/download>). 2021. 07. 10. 스크린샷.

[그림 1-10] 파이썬 개발 툴(Pycharm) 설치 파일 다운로드

(마) 파이썬 개발 툴을 설치하고 정상적으로 동작되는지 확인한다.

정상 설치 여부를 확인하기 위해 간단한 출력문을 실행하여 결과가 정상적으로 표시되는지 확인한다. 간단한 명령문을 입력하고 상단 우측의 붉은색 표시 버튼을 클릭하면 실행 결과가 하단에 표시된다.



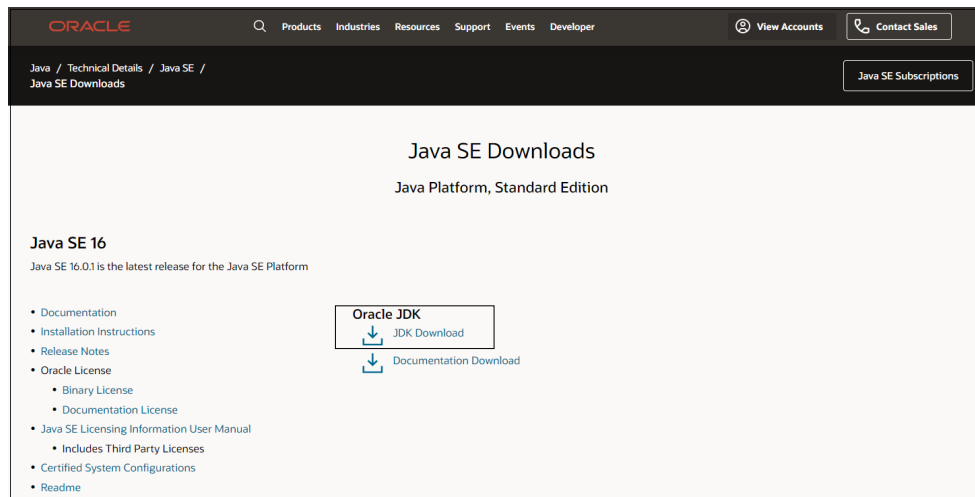
출처: 집필진 제작(2021)

[그림 1-11] Pycharm 간단한 명령문 입력 및 실행 결과 화면

(2) 자바 사용을 위한 개발환경을 구성한다.

(가) 오라클 공식 사이트에서 설치 파일을 다운받는다.

자바는 오라클 공식 사이트(<https://www.oracle.com/java/technologies/javase-downloads.html>)에서 사용자 환경에 맞는 설치 파일을 다운받아 설치한다.

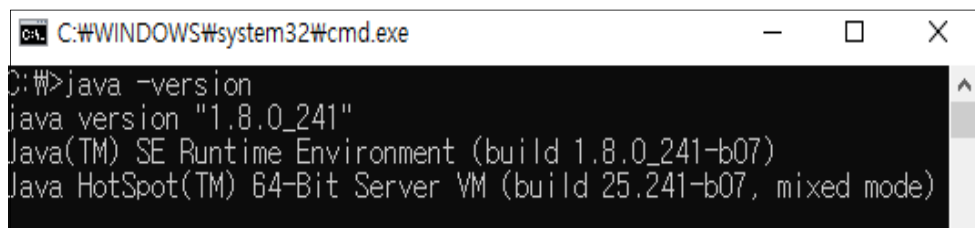


출처: JAVA 다운로드 사이트(<https://www.oracle.com/java/technologies/javase-downloads.html>). 2021. 07. 10. 스크린샷.

[그림 1-12] 자바 설치 파일 다운로드

(나) 다운로드 파일을 설치하고 정상적으로 동작하는지 확인한다.

정상 설치 여부를 확인하기 위해 명령어 화면에서 'java -version'을 설치한 자바의 버전이 정상적으로 출력되는지 확인한다.

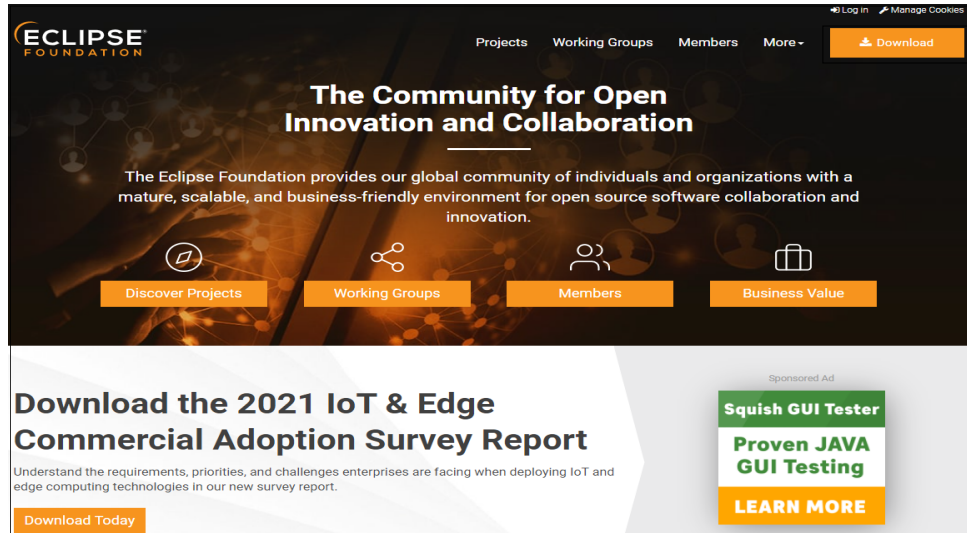


출처: 집필진 제작(2021)

[그림 1-13] 자바 설치 확인

(다) 자바 개발 툴을 다운로드한다.

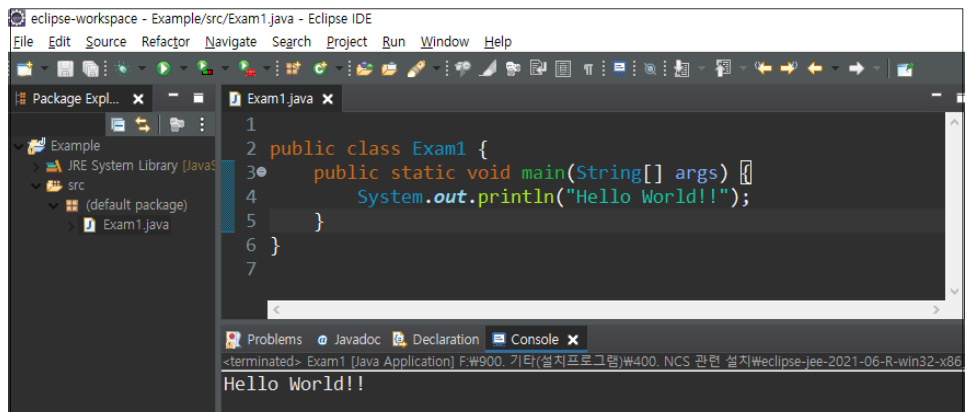
자바를 기반으로 한 IDE인 이클립스의 경우 <https://www.eclipse.org/> 에서 받을 수 있으며 환경에 맞는 설치 파일을 다운받아 설치한다.



출처: 이클립스 다운로드 사이트(<https://www.eclipse.org/>). 2021. 07. 10. 스크린샷.
[그림 1-14] 자바 개발 툴(Eclipse) 설치 파일 다운로드

(라) 다운로드 파일을 설치하고 정상적으로 동작되는지 확인한다.

정상 설치 여부를 확인하기 위해 간단한 화면 출력 코드를 작성하여 정상적으로 동작하는지 확인한다.



출처: 집필진 제작(2021)
[그림 1-15] 자바 개발 툴(Eclipse) 실행 화면

2. 언어의 특성을 적용하여 애플리케이션을 구현한다.

프로그래밍 언어에 따라 변수 선언 및 반복문, 조건문, 연산자 등의 표기 방식이 상이하니 해당 언어의 특성을 적용하여 애플리케이션을 구현한다. 프로그래밍 코딩 가이드가 존재하는 경우 가이드를 참고하여 코딩을 진행하고, 코드 인스펙션 등을 통해 가이드 준수 여부를 검토하여 최적화를 수행한다.

(1) 프로그램 명명 규칙을 파악한다.

프로그램 명명 가이드 등을 참고하여 파일명의 대소문자, 글자 수, 패키지 생성 규칙 등을 파악하여 신규 파일을 생성한다.

3.1.2 File Name

모든 파일의 이름은 64 글자를 넘지 않는 것으로 한다.

※ 참고

8.3 byte : ISO9660 표준

21 byte : Unix에서 'ls -l'로 한 줄에 딱 맞게 표시하기 위한 Size

64 byte : juliet - Unicode 지원

128 byte : romeo - 호환성 좋지 않음

3.2 Java/JSP

3.2.1 기본 구조

Java 와 JSP 파일 명은 기본적으로 다음과 같은 구조를 갖는다.

[A] [BBB] [CCC...C] [D] [E].[F..F]

파일의 확장자를 기술하며 java, JSP 가 될 수 있음.

인터페이스나 추상화 클래스인 경우에만 대문자 I 혹은 A로 표시한다. 일반 클래스는 아무것도 표시하지 않음.

역할 단위 명의 구분자를 대문자로 사용함.

파일의 구체적인 역할을 기술함. 단어의 연결로 구성하며 단어의 첫 글자만 대문자로 함.

업무 단위명의 3 자리 영문 약어로 모두 대문자로 기술함. 필요에 따라 업무 그룹이나 구분자가 올 수도 있음.

시스템을 구분할 수 있는 구분자로 대문자이며 본 문서 2.1.1에 기술된 내용을 참고함.

주) class 명은 의미있는 단어별로 대문자로 시작한다.

출처: 집필진 제작(2021)
[그림 1-16] 프로그램 명명 가이드 예시

(2) 주석 작성 방법 등에 대해 확인한다.

프로그래밍 언어의 문장 종료에 대한 표시 방법과 주석문 작성 방법에 대해 파악한다.

<표 1-17> 언어별 문장 종료 및 주석문 표시 방법

구분	JAVA	Python
문장 끝	- ;(세미콜론) 필요 System.out.println(1+2);	- ;(세미콜론) 생략 가능 print (1+2) print (1+2);
주석(한 줄)	//한 줄 주석	# 한 줄 주석
주석(여러 줄)	/* 여러 줄 주석 */	- 큰따옴표"""~""", 작은따옴표'~' 모두 가능 """ 여러 줄 주석 """

(3) 프로그램 코딩 규칙을 파악한다.

프로그래밍 코딩 가이드 등을 참고하여 들여쓰기 및 줄 나누기, 메소드, 조건문 등에 대한 작성 규칙을 파악하고, 변수 선언 방법에 대해 확인한다.

- 모든 내용은 들여쓰기를 적용하며 들여쓰기는 호환성을 고려하여 빈칸 4 컬럼으로 한다.
- 한 라인은 80컬럼을 넘지 않도록 한다.
- 모든 구성요소는 앞뒤로 한 칸의 공백을 둔다. (예외 : 단항 연산자, i++ 등)
- 한 구문에서 여러 대입식을 사용하지 않는다.

3.1.1.2 줄 나누기

한 행에 모두 기술하지 못할 경우 행을 나누는 규칙은 다음과 같다.

예시

- 콤마(,) 뒤에서 나눈다.
- 연산자 앞에서 나눈다.
- 나누어진 줄이 같은 단계라면 위의 줄과 같은 컬럼에 위치시킨다.

(예 1) Method 정의 혹은 호출 - Parameter 가 많을 경우 한 줄에 쓸 수 없다면, 한 줄에 하나씩 여러 줄에 기입한다. 한 줄에 모두 쓸 수 있는 경우라면 한 줄에 쓰도록 한다.

```
public DamageInfo getDamageDetail( String pDisasterMgmtNo,
                                   String pOccurAddrCode,
                                   String pDmgCauseCode,
                                   )throws SQLException, InformException {
```

(예 2) 긴 String 의 연결은 StringBuffer Class 를 이용한다.

```
StringBuffer sql1 = new StringBuffer("SELECT COLUMN1, COLUMN2, COLUMN3,
COLUMN4");
sql1.append("FROM TABLE1, TABLE2, TABLE3");
```

(예 3) 조건이 복잡한 비교 문장은 적당한 조건단위로 나누어 여러 줄에 사용한다.

```
If ( ( condition1 && condition2 )
    || (condition3 && condition4)
    || !(condition5 && condition6)
    ) {
    doSomethingAboutIt();
}
```

출처: 집필진 제작(2021)

[그림 1-17] 프로그램 코딩 가이드 문서 예시

(4) 변수 선언 방식과 선언 규칙에 대해 파악한다.

프로그래밍 언어마다 변수 선언 방식과 대소문자 구분 여부, 선언 규칙에 대한 차이가 있으므로 변수 선언 방법에 대해 확인한다.

〈표 1-18〉 언어별 변수 선언 방식 예시

구분	JAVA	Python
특징	1. _ 혹은 \$ 혹은 영문자로 시작 2. 자바 예약어(데이터 타입, 제어문, 반복문 등) 사용 불가 3. 길이에 제한 없고 자료형 지정 필요	1. _ 혹은 영문자로 시작 2. 특수문자 및 공백 사용 불가 3. 파이썬 예약어(for, while 등) 사용 불가 4. 자료형 생략 가능
예시(가능)	<pre>int iCnt = 100; String strName = "Exam";</pre>	<pre>iCnt = 1 strName = "Exam"</pre>
예시(불가능)	<pre>int 1Cnt = 100; String my*Name = "Exam";</pre>	<pre>for = 7 str^Name = "Exam"</pre>

(5) 조건문과 반복문의 적용 방법을 파악한다.

프로그래밍 언어에 따라 조건문과 반복문을 작성하는 방법과 구조가 차이가 있으므로 해당 언어의 조건문과 반복문 작성 방법을 파악한다.

(가) 조건문의 적용 방법을 파악한다.

조건문 작성 시 언어에 따라 블록에 대한 표시 방법과 다중 조건에 대한 처리 방법에 대한 차이를 파악한다.

〈표 1-19〉 언어별 조건문 적용 방법 예시

구분	JAVA	Python
특징	1. 판단 조건은 () 안에 표시하고 조건문 블록을 { }로 구별 2. 복수조건 else if 명령어 사용	1. 조건문 뒤에 :(콜론)을 사용하여 표시 및 구분 2. 복수조건 elif 명령어 사용
단순조건문	<pre>iCnt = 100; if(iCnt > 50){ System.out.println("big"); }else{ System.out.println("small"); }</pre>	<pre>iCnt = 100 if iCnt > 50 : print("big") else: print("small")</pre>
	실행 결과: big	
복수조건문	<pre>iCnt = 70; if(iCnt > 80){ System.out.println("very big"); }else if(iCnt > 50){ System.out.println("big"); }else { System.out.println("small"); }</pre>	<pre>iCnt = 70; if iCnt > 80 : print("very big") elif iCnt > 50 : print("big") else: print("small")</pre>
	실행 결과: big	

(나) 반복문의 적용 방법을 파악한다.

해당 프로그래밍 언어에서 사용하는 반복문의 블록 표시 방법과 여러 가지 반복문의 작성 방법을 파악한다.

〈표 1-20〉 언어별 반복문 적용 방법 예시

구분	JAVA	Python
for 반복문	<pre>for(int i =0; i < 3; i++){ System.out.println(i); }</pre>	<pre>for i in range(3): print(i)</pre>
	실행 결과: 0, 1, 2	
while 반복문	<pre>int i = 0; while(i <= 2){ System.out.println(i); i = i +1; }</pre>	<pre>i = 0 while i <= 2: print(i) i = i +1</pre>
	실행 결과: 0, 1, 2	

(6) 배열에 대한 적용 방법을 파악한다.

여러 개의 데이터를 저장, 관리할 수 있는 배열(파이썬의 경우 리스트)의 선언 및 처리 방법에 대해 파악한다. 자바의 경우 동일한 자료형만 저장 가능하지만 파이썬의 경우 서로 다른 자료형을 포함할 수 있다.

〈표 1-21〉 언어별 배열 적용 방법 예시

구분	JAVA	Python
배열(리스트) 선언	<pre>int[] arr = {1, 2, 3};</pre>	<pre>flist = [1, "String", 3]</pre>
배열(리스트) 출력	<pre>for (int i = 0; i < arr.length; i++) { System.out.println(arr[i]); }</pre>	<pre>for i in flist: print(i)</pre>
	결과: 1, 2, 3	결과: 1, String, 3

(7) 변수, 연산자, 반복 및 제어문 등을 이용하여 애플리케이션을 구현한다.

프로그래밍 언어에 따른 변수 및 연산자, 반복문, 제어문 등의 특성을 고려하여 애플리케이션을 구현한다.

(8) 애플리케이션을 수행하고 오류가 식별되는 경우 디버깅을 수행한다.

작성된 코드를 수행하여 문법 오류는 없는지 확인하고 애플리케이션을 수행하여 오류가 발생하는 경우 디버깅을 통해 오류를 해결한다.

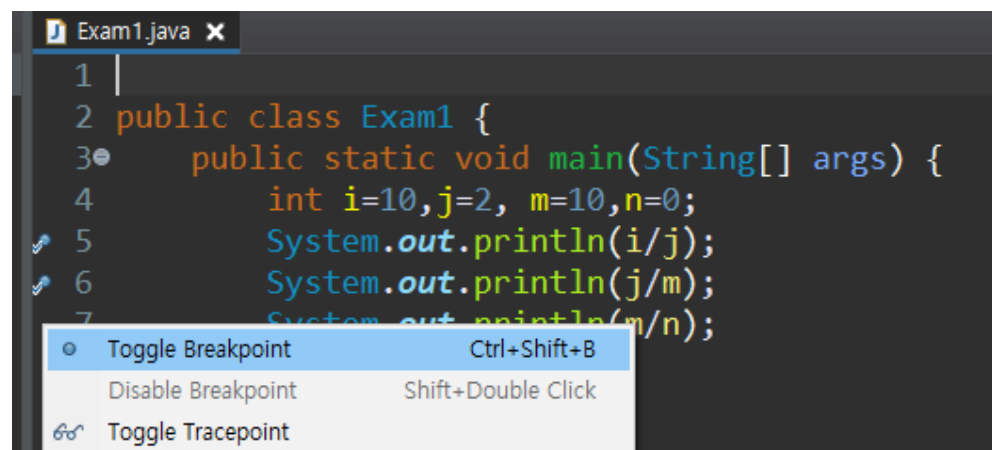
(가) 임의로 오류가 발생하는 코드를 작성한다.

〈표 1-22〉 자바 오류 발생 코드 예시

오류 발생 코드	오류 메시지
<pre>public class Exam1 { public static void main(String[] args) { int i=10,j=2, m=10,n=0; System.out.println(i/j); System.out.println(j/m); System.out.println(m/n); } }</pre>	<pre>Exception in thread "main" java.lang.ArithmeticException: / by zero at Exam1.main(Exam1.java:7)</pre>

(나) 오류가 의심되는 부분에 브레이크 포인트를 지정한다.

이클립스의 경우 소스코드 좌측의 라인에서 오류가 의심되는 부분을 더블클릭하거나 ‘마우스 우측 버튼 > Toggle Breakpoint’를 선택하여 디버깅 모드의 활용이 가능하다.



출처: 집필진 제작(2021)

[그림 1-18] 디버깅을 위한 브레이크 포인트 지정 방법

(다) 디버그 모드를 실행한다.

상단 바에 있는 벌레모양 Debug를 선택하거나 ‘Run > Debug’ 메뉴를 선택한다.

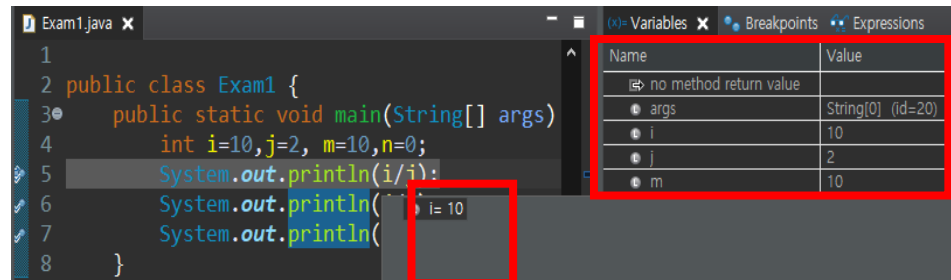


출처: 집필진 제작(2021)

[그림 1-19] 디버그 모드 실행 방법

(라) 단계별로 수행하면서 오류 원인을 확인한다.

디버깅 모드를 실행한 후 변수에 커서를 가져다 대거나 혹은 우측의 Variables 창을 통해 각 변수에 정확한 값이 들어가 있는지 확인하여 오류가 존재하는 부분을 수정한다.



출처: 집필진 제작(2021)

[그림 1-20] 디버깅 모드를 통한 오류 원인 확인

(마) 테스트를 수행한다.

여러 가지 테스트 케이스를 수행하고 오류가 발생하는 경우 디버깅을 통해 원인 확인 및 조치를 진행한다.

② 프로그래밍 언어의 특성을 활용하여 최적화를 수행한다.

프로그래밍 언어의 특성을 활용한 디자인 패턴, 보안, 성능, 정적/동적 테스트 등을 통해 애플리케이션에 대한 최적화를 수행한다.

1. 일반적인 프로그래밍 최적화 기법을 활용하여 최적화를 수행한다.

(1) 비트 연산 사용이 가능한 경우 산술 연산을 비트 연산으로 변경한다.

일반적으로 산술 연산과 대비하여 비트 연산이 더 빠르게 동작하므로 비트 연산으로 변경하여 최적화를 수행한다.

〈표 1-23〉 산술 연산과 비트 연산 비교

구분	산술 연산	비트 연산
더하기(+1)	i += 1;	i++;
빼기(-1)	i -= 1;	i--;
곱하기(*2)	i *= 2;	i <<= 1;
나누기(/2)	i /= 2;	i >>= 1;

(2) 공통 코드를 중복 실행하는 경우 한번 실행될 수 있도록 변경한다.

불필요한 수식을 제거하여 공통된 코드의 경우 한번 실행된 값을 활용할 수 있도록 변경하여 최적화를 수행한다.

〈표 1-24〉 공통 코드에 대한 최적화 예시

최적화 수행 전	최적화 수행 후
<pre>a = speed * time + k; b = speed * time * m; c = speed * time / n;</pre>	<pre>tmpData = speed * time; a = tmpData + k; b = tmpData * m; c = tmpData / n;</pre>

(3) 반복문 안에 변하지 않는 값이 반복 수행되지 않도록 변경한다.

반복문 안에서 동일한 결과값에 대한 불필요한 연산이 반복되는 경우 해당 코드를 반복문 밖으로 이동하여 최적화를 수행한다.

〈표 1-25〉 반복문 안에서의 불필요한 연산 제거를 통한 최적화 예시

최적화 수행 전	최적화 수행 후
<pre>int[] arr = {1,2,3,4}; int height=2,length=3,depth=4; for(int i = 0; i < arr.length ; i++){ arr[i] *= height * length * depth; }</pre>	<pre>int[] arr = {1,2,3,4}; int height=2,length=3,depth=4; int tmp = height * length * depth; for(int i = 0; i < arr.length ; i++){ arr[i] *= tmp; }</pre>

(4) 수행 작업에 대한 알고리즘 최적화를 진행한다.

수식 및 알고리즘이 존재하는 경우 반복문 사용 등을 지양하고 해당 알고리즘으로 변경하여 최적화를 수행한다.

〈표 1-26〉 알고리즘을 통한 최적화 예시

최적화 수행 전	최적화 수행 후
<pre>for(int i = 1; i <= Cnt ; i++) { SUM += i; }</pre>	<pre>SUM = (Cnt * (Cnt + 1)) / 2;</pre>

(5) api를 이용한 최적화를 진행한다.

프로그래밍 언어에서 제공하는 동일한 기능의 api가 존재하는 경우 api를 사용하여 최적화를 수행한다.

〈표 1-27〉 표준 API를 통한 최적화 예시

최적화 수행 전	최적화 수행 후
<pre>int arr[] = {5,7,10,14,9}; int imax = 0; for (int i = 1; i < arr.length; i++) { if (arr[i] > imax) { imax = arr[i]; } }</pre>	<pre>import java.util.Arrays; int arr[] = {5,7,10,14,9}; Arrays.sort(arr); imax = arr[arr.length-1];</pre>

(6) 문자열 연산 시 String 대신 StringBuilder를 이용하여 최적화를 수행한다.

String은 불변(Immutable)의 특성을 가지고 있어 문자열이 할당된 메모리 공간이 변하지 않아 '+' 연산자 등을 이용하여 문자열을 추가하는 경우 새로운 객체를 생성하여 시간과 자원을 사용하게 된다. 다만, 동기화가 필요한 경우라면 StringBuffer를 사용한다.

〈표 1-28〉 문자열 연산에 대한 최적화 예시

최적화 수행 전	최적화 수행 후
<pre>int Cnt = 10000; String str = ""; for (int i = 1; i <= Cnt ; i++) { str += "string" + i; }</pre>	<pre>int Cnt = 10000; StringBuilder builder = new StringBuilder(); for (int i = 1; i <= Cnt ; i++) { builder.append("string").append(i); }</pre>

(7) 최적화 수행 후 수행시간을 측정하여 최적화 여부를 확인한다.

최적화 수행 전과 수행 후의 프로그램 수행시간을 측정하여 실제로 최적화되었는지 여부를 확인한다. 수행시간은 프로그램 시작되는 지점과 끝나는 지점에서 시스템의 시간 차이를 기준으로 측정할 수 있으며, 측정 결과는 사용자 환경에 따라 변하거나 차이가 발생할 수 있다.

〈표 1-29〉 수행시간 측정을 통한 최적화 여부 검토

수행시간(적용 전)	수행시간(적용 후)
<pre>int Cnt = 100000; long start = System.nanoTime(); String str = ""; for (int i = 1; i <= Cnt ; i++) { str += "string" + i; } long end = System.nanoTime(); long duration = end - start; System.out.println(duration / 1000000000.0);</pre>	<pre>int Cnt = 100000; long start = System.nanoTime(); StringBuilder builder = new StringBuilder(); for (int i = 1; i <= Cnt ; i++) { builder.append("string").append(i); } long end = System.nanoTime(); long duration = end - start; System.out.println(duration / 1000000000.0);</pre>
결과: 8.9792875	결과: 0.0076047

2. 프로그래밍 언어에 적용 가능한 패턴을 확인하여 적용한다.

프로그래밍 과정에서 지속적으로 문제가 되었던 부분을 해결하고 최적화하기 위하여 여러 가지 패턴이 제시되었다. 애플리케이션 구현 시 적용 가능한 패턴을 확인하여 적용한다.

(1) 프로그래밍 언어의 특성을 활용하여 패턴을 적용한다.

프로그래밍 언어의 특성에 맞게 최적화를 위한 패턴을 적용한다. 디자인 패턴은 다양한 유형이 존재하므로 대표적인 패턴의 동작 및 구현 방식을 이해하고 적용 여부를 판단한다.

(가) 싱글톤(Singleton) 패턴을 적용한다.

설정 파일의 특정한 값과 같이 동일한 인스턴스를 참조할 때 하나의 인스턴스만 생성하여 처리함으로써 자원을 효율적으로 사용할 수 있는 장점이 있다. 소스코드를 참조하여 하나의 인스턴스로 처리가 가능한 경우에는 싱글톤 패턴을 적용한다.

〈표 1-30〉 싱글톤 패턴 적용 예시

파일	소스코드
Singleton.java	<pre> public class Singleton { private static Singleton instance; private Singleton(){} public static Singleton getInstance() { if(instance == null){ System.out.println("create instance"); instance = new Singleton(); } return instance; } } </pre>
SingletonTest.java	<pre> public class SingletonTest { public static void main(String[] args) { Singleton instance1 = Singleton.getInstance(); Singleton instance2 = Singleton.getInstance(); System.out.println("instance1/instance2 : " + (instance1 == instance2)); } } </pre>
결과값	<pre> create instance instance1/instance2 :true </pre>
적용 장점	<p>서로 다른 Class에서 instance를 호출하여도 최초 1회만 실행되며 반환된 인스턴스의 값은 동일하다. 따라서 동일한 데이터가 설정 파일이나 DB 조회를 통해 빈번하게 참조되는 경우에는 싱글톤 패턴을 통해 1회만 수행한 후 동일 인스턴스를 사용함으로써 자원 사용율을 향상시킬 수 있다.</p>

(나) 스트래티지 패턴(Strategy Pattern)

사용자가 자신에게 맞는 전략(Strategy)을 취사선택하여 로직을 수행할 수 있는 방법이다. 향후 확장성에 대한 고려가 필요한 경우 소스코드를 참조하여 스트래티지 패턴을 적용한다.

〈표 1-31〉 스트래티지 패턴 적용 예시

파일	소스코드
Animal.java	<pre>public interface Animal { public void printName(); }</pre>
AnimalCall.java	<pre>public class AnimalCall { private Animal animal; public void setAnimal(Animal animal) { this.animal = animal; } public void callName(){ animal.printName(); } }</pre>
Cat.java	<pre>public class Cat implements Animal{ public void printName() { System.out.println("Cat!!"); } }</pre>
Dog.java	<pre>public class Dog implements Animal{ public void printName() { System.out.println("Dog!!"); } }</pre>
StrategyTest.java	<pre>public class StrategyTest { public static void main(String[] args) { AnimalCall anicall = new AnimalCall(); anicall.setAnimal(new Dog()); anicall.callName(); anicall.setAnimal(new Cat()); anicall.callName(); } }</pre>
결과값	<pre>Dog!! Cat!!</pre>
적용 장점	<p>전략 패턴의 가장 큰 장점은 확장성으로 Animal 인터페이스를 상속해서 같은 기능(책임)만 수행할 수 있다면 코드 영향 범위를 최소화하면서 얼마든지 다른 동물도 추가할 수 있다.</p>

(2) 기타 응용소프트웨어 구현을 위해 사용 가능한 패턴을 적용한다.

다른 디자인 패턴의 내용을 참고하여 패턴 적용이 가능한 경우 해당 패턴을 적용하여 소스코드를 최적화한다.

〈표 1-32〉 디자인 패턴의 종류 및 설명

구분	종류	설명
생성 패턴 (Creational Patterns)	Factory Method	어떤 클래스의 인스턴스를 만들지 서브클래스에서 결정하도록 책임을 위임하는 패턴
	Abstract Factory	관련 있는 객체들을 모아서 팩토리로 만들고 조건에 따라 팩토리 중에서 선택하게 하는 패턴
	Builder	객체의 생성 단계를 캡슐화하여 구축 공정을 동일하게 이용하도록 하는 패턴
	Prototype	원형이 되는 객체를 복제하여 새로운 객체를 생성하는 패턴
	Singleton	인스턴스를 한 번만 생성하여 메모리에 저장하여 사용함으로써 하나의 인스턴스를 보장하는 패턴
구조 패턴 (Structural Patterns)	Adapter	서로 다른 인터페이스를 가진 클래스를 함께 사용할 수 있도록 하는 패턴
	Bridge	기능과 구현을 분리하여 독립적으로 변형과 확장이 가능하도록 결합도를 낮춘 패턴
	Composite	여러 개의 객체들로 구성된 복합 객체와 단일 객체를 클라이언트에서 동일하게 사용하는 패턴
	Decorator	객체에 추가적인 기능을 유연하게 확장하는 패턴
	Façade	복잡한 클래스들을 묶어 통합된 인터페이스를 제공하는 패턴
행위 패턴 (Behavioral Patterns)	Proxy	실제 객체가 아닌 가상 객체를 통해 기능을 처리하는 패턴
	Interpreter	문법 규칙을 정의하고 해석하는 패턴
	Template	공통 메소드를 상위 클래스에서 정의, 세부 사항은 하위 클래스에서 구현하는 패턴
	Command	요청을 캡슐화하여 여러 기능을 실행할 수 있는 패턴
	Iterator	컬렉션의 구현은 노출시키지 않고 요소들에 접근할 수 있는 패턴
	Mediator	객체들 간의 상호작용 행위를 캡슐화하여 관리하는 패턴
	Memento	객체의 상태정보를 저장하고 원하는 시점의 이전 상태로 복원할 수 있는 패턴
	Observer	객체의 상태가 업데이트되면 객체에 의존하는 다른 객체에 알리고 자동으로 내용을 갱신하는 패턴
	Strategy	행위를 클래스로 캡슐화하여 필요에 따라 동적으로 대체가 가능하도록 한 패턴
	Visitor	데이터와 구조를 분리하여 구조를 수정하지 않고 새로운 기능을 추가할 수 있는 패턴

3. 프로그래밍 언어 특성에 따른 보안 취약점을 검토하고 최적화한다.

보안 취약점 검토를 위하여 시큐어 코딩(Secure Coding)이나 OWASP 등의 취약점을 분석하여 최적화한다.

(1) 입력 데이터 검증 및 표현 측면의 보안 약점을 확인하고 최적화한다.

(가) SQL 삽입 공격에 대한 보안 취약점을 검토하고 제거한다.

공격자가 입력 form 및 URL 입력란 등에 SQL문을 삽입하여 DB를 조회하거나 조작할 수 있으므로, preparedStatement를 사용하고 입력값에 대해 SQL 구문 제한, 특수문자 제한, 길이 제한 등의 필터링을 적용한다.

(나) 경로 조작 및 자원 삽입 공격에 대한 보안 취약점을 검토하고 제거한다.

검증되지 않은 입력값을 허용하는 경우 악의적인 코드가 실행되거나 시스템 자원에 무단 접근이 가능하므로, 자원 식별자(IP, Port, Directory 등)와 같은 중요 정보는 외부에서 입력받지 않고 처리하며, 다른 경로 파일을 접근이 가능할 수 있는 위험한 문자열(ex: ", /, \)을 제거한다.

(다) 파일 업로드에 대한 보안 취약점을 검토하고 제거한다.

파일의 확장자와 형식에 대한 검증 없이 업로드하는 경우 .exe 파일과 같은 형식의 실행 파일은 보안 위협이 될 수 있으므로 확장자와 형식을 확인하여 업로드가 가능하도록 개선한다.

(라) 정수 오버플로에 대한 보안 취약점을 검토하고 제거한다.

정수형 변수에 저장된 값이 허용된 정수값 범위를 벗어나 프로그램이 예기치 않게 동작 가능한 보안 취약점으로, data type의 범위에 있는지 사전에 체크하도록 개선한다.

(2) 보안 기능 측면의 보안 약점을 확인하고 최적화한다.

(가) 중요 정보를 적절한 인증 없이 조회하거나 변경할 수 있는 취약점이 존재하는지 확인하고 인증을 추가한다.

(나) 개인정보 및 비밀번호 등의 중요 정보를 전송하거나 저장 시 암호화하지 않아 정보가 노출 가능한 취약점이 존재하는지 확인하고 암호화하여 처리한다.

(다) 소스코드에 중요 정보(암호화 키, 비밀번호 등)를 직접 코딩하는 경우 소스코드 유출 시 중요 정보가 노출되고 주기적 변경이 어려운 취약점이 존재하는지 확인하고, 중요 정보는 별도의 설정 파일에 관리하도록 개선한다.

(라) 인증 시도 수를 제한하지 않아 공격자가 반복적으로 임의 값을 입력하여 계정 권한을 획득 가능한 취약점이 존재하는지 확인하고 인증 시도를 제한하도록 개선한다.

(3) 에러 처리, 코드 오류 등에서 발생할 수 있는 보안 약점을 확인하고 최적화한다.

(가) 오류 메시지나 스택 정보에 시스템 내부구조가 포함되어 민감한 정보, 디버깅 정보가 보안 약점 메시지를 통해 노출되는 취약점이 존재하는지 확인하여 개선한다.

(나) 시스템 오류 상황을 처리하지 않아 프로그램 실행 정지 등 의도하지 않은 상황이 발생 가능한 취약점이 존재하는지 확인하여 개선한다.

4. 언어 특성에 따라 발생할 수 있는 성능과 가용성 측면의 문제를 확인하여 최적화를 수행한다.

(1) 데이터 전송 속도를 확인하여 최적화한다.

서버 간의 데이터 전송 시 프로그래밍 언어의 특성을 적절하지 않게 반영하였거나, 코드 작성이 잘못된 부분은 없는지 확인하여 속도에 문제가 발생하지 않도록 개선한다.

(2) 클라이언트 처리속도를 확인하여 최적화한다.

클라이언트 설치 파일 또는 웹 UI의 로딩 속도를 확인하여, 필요하지 않거나 용량이 큰 파일의 다운로드로 인해 속도가 저하되는지 확인하여 개선한다.

(3) 데이터베이스로부터 조회되는 속도를 확인한다.

데이터베이스 내에서 조회하고자 하는 내용이 적절한 시간 내에 응답하는지 확인하고, 프로그래밍 언어의 특성을 적절하지 않게 반영한 부분이 있다면 분석하여 보완한다.

(4) 가용 자원의 현황과 프로그램 구동 시의 점유율을 확인하여 최적화한다.

프로그램 실행 시 CPU, 메모리, 디스크 용량의 자원 상황과 점유율을 확인하여 불필요한 코드는 삭제하고 최적화를 수행한다.

(가) CPU의 자원 상황과 점유율이 높은 경우 CPU를 많이 사용하는 기능에 대해 연산, 조건, 반복, 암/복호화 등의 코드를 검토 후 최적화한다.

(나) 메모리의 자원 상황과 점유율을 확인하여 메모리를 많이 사용하는 기능에 대해 자원 반환, 연산, 조건, 불필요한 객체 제거 등에 대한 검토 후 최적화한다.

(다) 디스크 용량의 자원 상황과 점유율을 확인하여 디스크를 많이 사용하는 기능에 대해 불필요한 로그나 파일 생성 등을 검토 후 최적화한다.

5. 소스코드 품질 측면에서의 최적화를 수행한다.

(1) 프로그램 명명 가이드를 기준으로 최적화를 수행한다.

(가) 패키지명명 규칙 표준 준수 여부를 확인한다.

프로젝트 표준 및 업무 기능에 맞게 패키지 구조가 정의되고 해당 프로그램의 패키지 구조가 생성 규칙을 준수하였는지 확인한다.

(나) 파일명 명명 규칙 표준 준수 여부를 확인한다.

업무 기능 및 대문자/소문자 사용 규칙 등이 파일명 생성 규칙을 준수하였는지 확인한다.

(다) 변수명 명명 준수 여부를 확인한다.

변수에 대한 Prefix 및 길이, 대소문자 규칙 등이 변수명 생성 규칙을 준수하였는지 확인한다.

(라) 메소드 명명 준수 여부를 확인한다.

메소드 길이 및 대소문자 규칙 등이 메소드 생성 규칙을 준수하였는지 확인하고 가이드와 상이한 경우 최적화한다.

(2) 프로그램 코딩 가이드를 기준으로 최적화를 수행한다.

(가) 주석문의 가이드 준수 여부를 확인한다.

한 줄 및 여러 줄, 프로그램 설명 등에 대한 주석문이 작성 규칙을 준수하였는지 확인한다.

(나) 들여쓰기의 가이드 준수 여부를 확인한다.

공백이나 Tab 문자를 이용한 들여쓰기가 작성 규칙을 준수하였는지 확인하고 가이드와 상이한 경우 수정한다.

(다) 선언문의 가이드 준수 여부를 확인한다.

변수 선언 및 초기화가 작성 규칙을 준수하였는지 확인하고 가이드와 상이한 경우 수정한다.

(라) 블록(Block Statement)의 가이드 준수 여부를 확인한다.

if-else, for loop 등에 대한 블록문이 작성 규칙을 준수하였는지 확인하고 가이드와 상이한 경우 수정한다.

(3) 불필요한 코드가 존재하는지 파악하여 최적화를 수행한다.

(가) 필요하지 않은 코드가 존재하는지 확인하여 최적화를 수행한다.

기능과 무관한 부분, 사용하지 않는 변수 선언 및 초기화, 사용하지 않는 객체 생성 등 필요하지 않은 코드는 삭제한다.

(나) 중복된 코드가 존재하는지 확인하여 최적화를 수행한다.

동일한 Import문이나 중복된 코드가 있는지, 유사 데이터들의 그룹이 중복되지 않았는지 확인하여 수정한다.

(다) 복잡 코드를 확인하여 최적화를 수행한다.

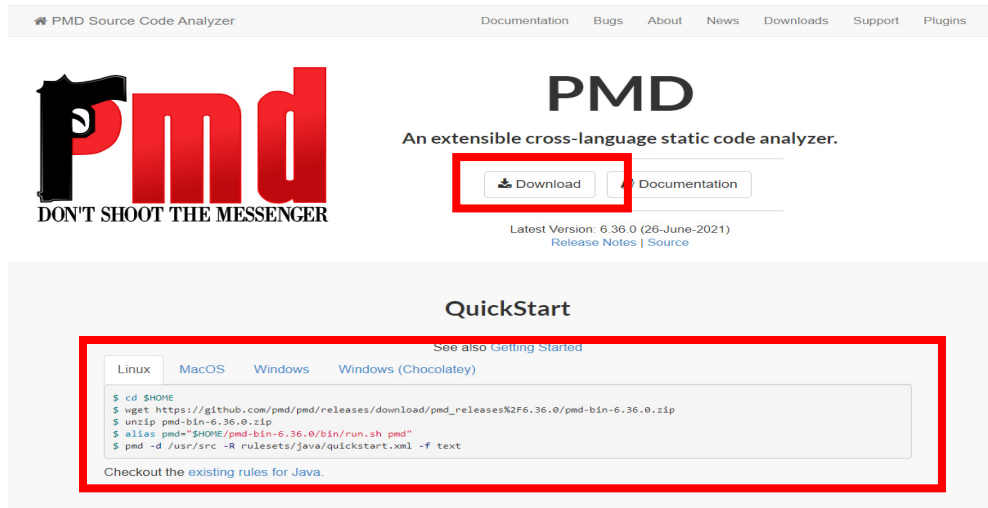
조건문이나 반복문이 많거나 코드의 양이 많은 복잡한 코드의 경우 별도의 파일이나 메소드로 분리한다.

(4) 정적 분석 도구를 활용하여 최적화를 수행한다.

소프트웨어를 실행하지 않고 코드 레벨에서 문제를 사전에 확인할 수 있는 정적 분석을 위한 오픈소스 코드 분석 도구로 기본 제공 규칙 세트가 포함되어 있고 사용자 지정 규칙을 작성할 수 있다.

(가) PMD를 다운로드한다.

PMD 사이트에서 압축 파일(.zip)을 다운받아 압축을 해제한다.



출처: PMD 사이트(<https://pmd.github.io/>). 2021. 07. 26. 스크린샷.
[그림 2-21] PMD 다운로드 및 사용법

(나) PMD를 실행한다.

압축 해제한 폴더에서 CMD 명령을 이용하여 PMD를 수행한다.

〈표 1-33〉 PMD를 이용한 최적화 수행 예시

구분	내용
점검 소스 및 내용	<ul style="list-style-type: none"> - 파일명: C:\src\Err1.java - 소스코드 <pre> import java.util.Date; public class Err1 { public static void main(String[] args) throws Exception { int my_num = 1; int iNum = 2; System.out.println(my_num); } } </pre>
PMD 수행	<pre> C:\pmd-bin-6.36.0\bin>pmd.bat -d c:\src -R rulesets/java/quickstart.xml -f text </pre>
PMD 수행 결과	

(다) PMD에서 식별된 룰셋을 검토하여 수정이 필요한 항목은 최적화를 수행한다.

식별된 룰셋의 내용을 확인하여 사용되지 않거나 중복된 항목은 제거하고, 개선이 필요한 항목은 레퍼런스 코드를 참고하여 수정한다. 관련 예시 코드는 <https://pmd.github.io/latest/> 에서 검색을 통해 확인이 가능하다.

〈표 1-34〉 룰셋 항목별 최적화 예시

룰셋	최적화 전	최적화 후
AvoidReassigningParameters	public void getID(String id) { id = id.trim();	public void getID(String id) { String trimmedID = id.trim();
EqualsNull	if (x.equals(null))	if (x == null) {
DuplicateImports	import java.lang.String; import java.lang.*;	import java.lang.String;
StringInstantiation	private String id = new String("test")	String bar = "test";
StringToString	String id = "test"; return id.toString();	String id = "test"; return id;
InefficientStringBuffering	StringBuffer sb = new StringBuffer("Dir "+System.getProperty("tmpDir"));	StringBuffer sb = new StringBuffer("Dir = "); sb.append(System.getProperty("tmpDir"));
UselessStringValueOf	public String convert(int i) { String s; s = "a" + String.valueOf(i);	public String convert(int i) { String s; s = "a" + i;
WhileLoopsMustUseBraces	while (true) x++;	while (true) { x++; }
UnnecessaryConversionTemporary	public String convert(int x) { String res = new Integer(x).toString(); return res;	public String convert(int x) { return Integer.toString(x);

6. 최적화가 완료된 프로그램 소스에 대한 테스트를 수행한다.

최적화 과정에서 개발자의 실수 등에 의해 오류가 발생할 수 있으므로 수정한 사항에 대해 테스트를 통한 점검을 수행한다.

(1) 단위 테스트를 수행한다.

최적화 작업 후 테스트가 가능한 최소 단위의 모듈에 대한 기능을 점검하여 정상 동작 여부를 확인하고 오류가 발생하면 수정한다.

(2) 통합 테스트를 수행한다.

단위 테스트가 완료된 기능을 통합하여 모듈 간의 상호 작용이 올바르게 동작하는지 점검한다.

(3) 회귀 테스트를 수행한다.

최적화 이전에 정상적으로 동작했던 기능들이 수정된 후에도 정상적으로 동작하는지 확인한다.

수행 tip

- 시큐어 코딩(Secure Coding)은 소프트웨어(SW)를 개발함에 있어 개발자의 실수, 논리적 오류 등으로 인해 SW에 내포될 수 있는 보안 취약점(Vulnerability)을 배제하기 위한 코딩 기법으로 관련 가이드는 한국인터넷진흥원 자료실(<https://www.kisa.or.kr/public/laws/laws3.jsp>)에서 다운로드할 수 있다
- PMD의 경우 전자정부 표준 프레임워크에서는 Code Inspection을 위한 룰셋으로 논리오류/구문오류/참조오류 영역을 대상으로 하는 총 39개의 룰을 표준으로 선정하고 있다.
- 프로젝트 상황 및 업무적 특성, 프레임워크 종류 등에 따라 PMD 룰셋에서 식별된 항목에 대한 처리 방법이 달라질 수 있으므로 모호한 항목의 경우 응용 아키텍터 및 품질 담당자와 협의하여 수정 여부를 결정한다.

교수 방법

- 프로그래밍 언어에 대한 분류 체계와 종류에 대해 설명한다.
- 프로그래밍 언어의 발전 과정과 각 언어의 특성을 설명한다.
- 프로그래밍 언어의 특성을 고려하여 애플리케이션 구현을 위한 환경 설정 및 구현 방법에 대해 설명한다.
- 프로그래밍 언어의 특성을 이용하여 상황에 따른 최적화 방법을 설명한다.

학습 방법

- 프로그래밍 언어의 분류 체계 및 주요 언어의 종류를 파악한다.
- 프로그래밍 언어의 특성을 설명하고, 잘못된 내용을 확인한다.
- 애플리케이션 구현을 위한 환경 설정 및 구현을 실습한다.
- 언어의 특성을 이용하여 최적화하는 방법을 실습한다.

평가 준거

- 평가자는 학습자가 학습 목표를 성공적으로 달성하였는지를 평가해야 한다.
- 평가자는 다음 사항을 평가해야 한다.

학습 내용	학습 목표	성취수준		
		상	중	하
언어별 특성 파악	- 프로그래밍 언어별 특성을 파악하고 설명할 수 있다.			
애플리케이션 구현 및 최적화	- 파악된 프로그래밍 언어의 특성을 적용하여 애플리케이션을 구현할 수 있다.			
	- 애플리케이션을 최적화하기 위해 프로그래밍 언어의 특성을 활용할 수 있다.			

평가 방법

- 평가자 질문

학습 내용	평가 항목	성취수준		
		상	중	하
언어별 특성 파악	- 프로젝트 성격에 적합한 프로그램 언어의 특성에 대한 숙지 여부			
	- 프로그래밍 언어에 대한 분류 체계와 각 분류별 언어의 종류에 대한 숙지 여부			
애플리케이션 구현 및 최적화	- 애플리케이션 최적화를 위한 대상 및 방법에 대한 숙지 여부			
	- 리팩토링 및 디자인 패턴, 시큐어 코딩의 개념 및 기법에 대한 숙지 여부			

• 평가자 체크리스트

학습 내용	평가 항목	성취수준		
		상	중	하
언어별 특성 파악	- 프로그래밍 언어의 발전 과정과 시대별 언어에 대한 숙지 여부			
	- 프로그래밍 언어의 개별 특성에 대한 숙지 여부			
애플리케이션 구현 및 최적화	- 프로그래밍 언어의 특성을 적용하여 애플리케이션을 구현하고 오류를 해결할 수 있는 능력			
	- 프로그래밍 언어의 특성을 활용하여 중복 코드 등의 제거를 통한 최적화 능력			

피드백

1. 평가자 질문

- 프로그래밍 언어의 특성에 대해 충분히 숙지하고 있는지 평가하고, 수정이 필요한 경우 관련 내용을 알려준다.
- 애플리케이션 최적화를 위한 검토 대상 및 최적화 기법에 대해 숙지하고 있는지 평가하고, 잘못된 내용이 있는 경우 알려준다.
- 디자인 패턴과 리팩토링의 개념 및 유형에 대해 활용이 가능하도록 안내한다.
- 학습자들의 프로그래밍 언어 사용 경험을 공유할 수 있는 기회를 제공한다.
- 평가자 질문 결과가 '상'인 경우 다른 학습자와 모범 사례를 공유하고 설명할 수 있도록 안내한다.
- 평가자 질문 결과가 '중'인 경우 부족한 부분을 보완할 수 있도록 설명한다.
- 평가자 질문 결과가 '하'인 경우 모범 사례를 참조하여 다시 학습할 수 있도록 유도한다.

2. 평가자 체크리스트

- 프로그래밍 언어의 발전 과정과 시대별 언어에 대한 지식을 충분히 숙지할 수 있도록 안내한다.
- 애플리케이션 구현 과정에서 오류가 발생하는 경우 다른 학습자와 협업하여 해결할 수 있도록 안내한다.
- 프로그래밍 언어의 특성을 고려한 최적화 방법과 최적화 사례에 대한 경험을 공유할 수 있는 기회를 제공한다.
- 평가자 체크리스트 결과가 '상'인 경우 다른 학습자와 모범 사례를 공유하고 설명할 수 있도록 안내한다.
- 평가자 체크리스트 결과가 '중'인 경우 부족한 부분을 보완할 수 있도록 설명한다.
- 평가자 체크리스트 결과가 '하'인 경우 모범 사례를 참조하여 다시 학습할 수 있도록 유도한다.

2-1. 라이브러리 선정

학습 목표

- 애플리케이션에 필요한 라이브러리를 선정할 수 있다.

필요 지식 /

① 라이브러리(Library)

라이브러리는 프로그램을 효율적으로 개발할 수 있도록 필요한 기능이 구현된 프로그램을 모은 집합체이며, 일반적으로 설치 파일과 도움말, 예시 코드 등을 제공한다.

1. 라이브러리 개념과 목적

라이브러리란 영어로 도서관을 의미하며 코드의 재사용 및 부품화, 기술 유출 방지를 위하여 하나 이상의 Subroutine 혹은 Function들의 집합으로 일반적으로 컴파일되어 실행이 가능한 Object Code 형태로 제공된다.

2. 라이브러리 구성

일반적으로 사용 방법을 위한 문서와 도움말, 예시 코드, 함수, 클래스, 값, 자료형 사양을 포함할 수 있다.

(1) 도움말: 라이브러리에 대한 설명 및 사용 방법 등에 대한 도움말 문서

(2) 설치 파일: 라이브러리를 연동하기 위해 제공되는 설치 파일

(3) 예시 코드: 애플리케이션 구현 시 라이브러리를 쉽게 연동할 수 있도록 제공되는 샘플 코드

② 라이브러리의 유형

1. 별도 설치 여부에 따른 유형

(1) 표준 라이브러리(Standard Library)

프로그래밍 언어와 함께 제공되는 라이브러리로 표준 라이브러리를 이용하면 별

도의 파일 설치 없이 다양한 기능을 이용할 수 있다. 각 프로그래밍 언어의 표준 라이브러리는 기본 기능 이외에 날짜와 시간 등의 다양한 추가 기능을 이용할 수 있는 API를 제공한다.

(2) 외부 라이브러리(3rd Party Library)

별도의 파일을 설치하여 사용할 수 있는 라이브러리를 의미한다. 누구나 개발하여 사용하거나 공유할 수 있고, 관련 커뮤니티나 인터넷 검색 등을 이용하여 공유된 라이브러리를 사용할 수 있다.

2. 코드 결합 방식에 따른 유형

(1) 정적 라이브러리(Static Library)

프로그램을 빌드할 때 라이브러리가 제공하는 코드를 실행 파일에 넣는 방식으로 컴파일러가 원시 소스를 컴파일할 때 참조되는 모듈이다. 동작 코드가 실행 바이너리에 포함되어 별도의 추가 작업 없이 독립적으로 라이브러리 함수 사용이 가능하다.

(2) 동적 라이브러리(Dynamic Library)

공통적으로 필요한 기능들을 프로그램과 분리하여 필요시에만 호출하여 사용할 수 있도록 만든 라이브러리를 의미한다. 필요할 때만 해당 라이브러리를 메모리로 불러들일 수 있어 실행 파일의 크기를 줄일 수 있고, 사용이 끝나면 메모리에서 제거되어 효율적인 메모리 사용이 가능하다. 윈도우에서는 주로 .dll 확장자, 리눅스에서는 주로 .so 확장자를 가진다.

③ 라이브러리, 프레임워크, 아키텍처, 플랫폼 비교

IT에서 일반적으로 사용되는 개발 용어인 프레임워크, 아키텍처, 플랫폼의 차이점과 예시를 통해 용어들 간의 상관관계를 확인할 수 있다.

〈표 2-1〉 라이브러리와 유사 용어 설명

항목	설명	예시
Library	코드 재사용 및 부품화를 위하여 필요한 기능에서 호출하여 사용할 수 있도록 제공되는 모듈의 집합	jQuery, DLL, Class, Jar 등
Framework	응용프로그램 표준 구조를 구현하기 위한 클래스와 라이브러리 모임	Spring, Django 등
Architecture	여러 가지 컴퓨터 구성요소들에 대한 전반적인 기계적 구조와 이를 설계하는 방법	모노리틱 아키텍처, MSA 등
Platform	여러 가지 기능을 제공해주는 프로그램 실행이 가능한 공통 실행환경으로 플랫폼 위에 다른 플랫폼이 존재할 수 있음	Window, Linux, JVM 등

수행 내용 / 라이브러리 선정하기

재료 · 자료

- 요구사항 관련 문서
- 개발 표준 관련 문서
- 프로그래밍 언어별 표준 라이브러리 도움말
- 프로그래밍 언어별 외부 라이브러리 종류

기기(장비 · 도구)

- 전산 장비: 인터넷, 컴퓨터, 프린터, 복사기, 빔 프로젝터
- 지원 도구: 문서 작성 도구, 문서 발표 도구, 소프트웨어 개발 도구

안전 · 유의 사항

- 프로그래밍 언어별로 제공되는 라이브러리 유형을 사전에 확인한다.
- 라이브러리 선정 시 지식 재산권 및 라이선스를 확인할 수 있도록 유의한다.
- 프로그래밍 언어와 버전, 실행환경 등을 고려하여 선정한다.
- 요구사항과 애플리케이션 기능에 적합한 라이브러리를 선택하도록 한다.

수행 순서

① 라이브러리 적용이 필요한 대상을 식별한다.

1. 애플리케이션 구현을 위한 요구사항을 분석한다.

애플리케이션의 기능 구현에 필요한 요구사항 내용을 확인하고 유형(기능성, 성능, 보안, 사용성 등)을 확인한다.

2. 업무 기능을 분석한다.

요구사항 분석 내용을 기반으로 업무 기능을 분류한다. 유사한 기능을 그룹화하고 누락되는 업무 기능이 없도록 정리한다.

3. 라이브러리 적용 여부를 검토한다.

업무적 특성 및 개발 시간, 노력, 기술 난이도 등을 고려하여 자체 개발 및 라이브러리 적용 여부 등에 대한 검토를 진행한다.

〈표 2-2〉 업무 기능에 따른 라이브러리 적용 검토 결과 예시

Level1	Level2	Level3	기능 설명	구현 방법
회원 관리	회원 가입	휴대폰 인증	휴대폰 인증을 통한 회원 가입 기능	솔루션 구매, 연동
		인증서 인증	(구)공인인증서를 이용한 회원 가입 기능	솔루션 구매, 연동
	회원 탈퇴	회원 탈퇴	기존 회원의 탈퇴 기능	자체 개발
관리자 메뉴	공지사항 관리	공지사항 등록	신규 공지사항 등록 기능	라이브러리 적용
		공지사항 수정	등록된 공지사항 수정 기능	라이브러리 적용
	대시보드	회원 현황	성별, 연령대별 현황을 그래프로 표시	라이브러리 적용

4. 라이브러리 적용이 필요한 기능을 정리한다.

업무 기능을 기반으로 자체적으로 개발하지 않고 라이브러리를 적용할 기능을 식별한다.

② 애플리케이션에 필요한 라이브러리를 검색한다.

1. 프로그래밍 언어에서 제공하는 표준 라이브러리를 검색한다.

일반적인 프로그래밍 언어의 경우 날짜 및 시간, 수학 연산 및 통계, 다양한 입출력, 멀티미디어 등과 같은 다양한 라이브러리를 제공한다.

〈표 2-3〉 일반적인 프로그래밍 언어의 표준 라이브러리 제공 기능

주요 기능	기능 설명
문자열 연산	일반적인 문자열의 조작(합치기, 자르기 등)을 처리
날짜 및 시간	현재 시각 및 날짜, 시간 등에 대한 처리
수학 모듈	수학 함수 및 통계 함수를 이용한 계산
운영체제	현재 디렉터리 등 운영체제 인터페이스 처리
파일/디렉터리	파일 조회 및 생성, 시스템 경로 조작
프로토콜	HTTP, SMTP, FTP 등의 프로토콜을 사용
멀티미디어	사운드 및 비디오 같은 멀티미디어 데이터를 처리

(1) 표준 라이브러리의 개념과 역할을 파악한다.

프로그래밍 언어와 함께 제공되는 라이브러리로 별도의 파일 설치 없이 날짜와 시간, 운영체제, 파일/디렉터리 처리 등의 기능을 이용할 수 있다.

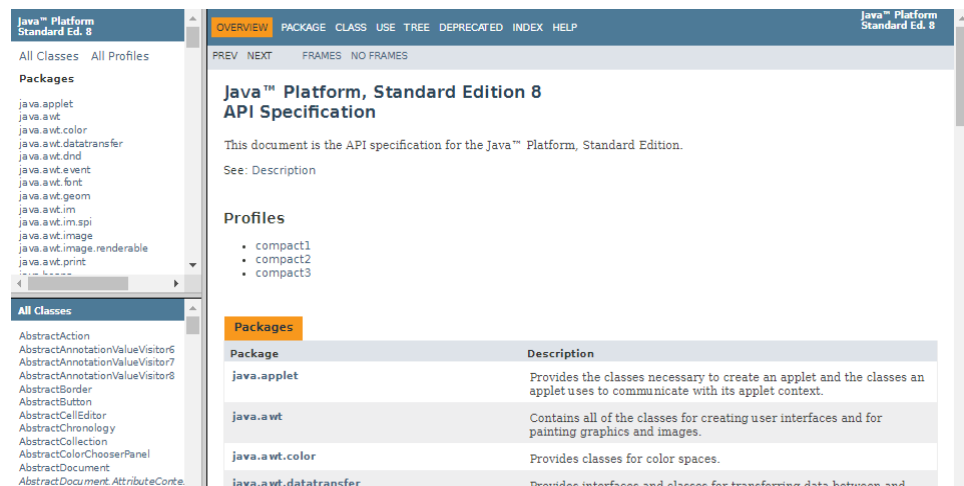
(2) 표준 라이브러리의 종류를 검색한다.

프로그래밍 언어에서 제공하는 표준 라이브러리를 검색하여 애플리케이션 구현에 적합한 라이브러리가 존재하는지 확인한다.

(가) 프로그래밍 언어의 표준 라이브러리 문서를 파악한다.

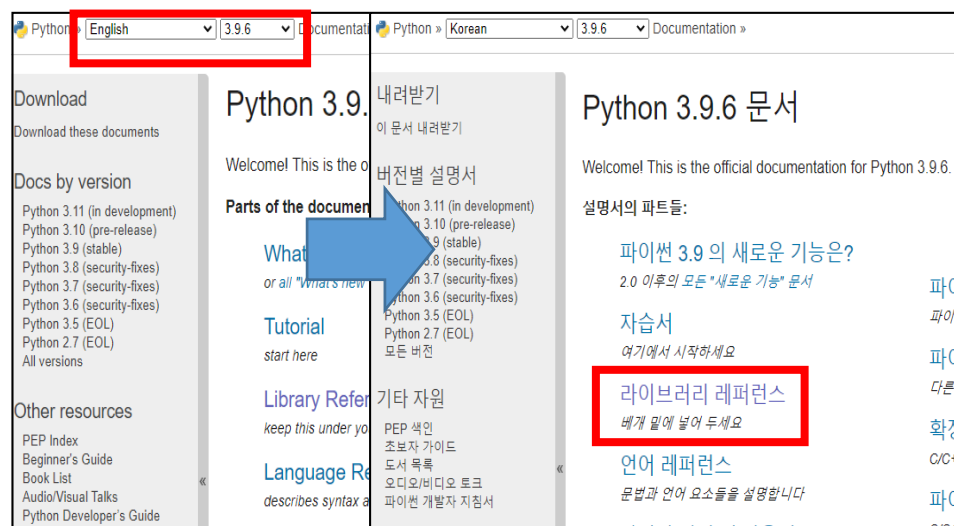
프로그래밍 언어에서 제공하는 표준 라이브러리에 대한 공식 사이트 및 문서를 확인한다.

1) 자바의 표준 라이브러리 문서를 확인한다.



출처: 자바 표준 라이브러리(<https://docs.oracle.com/javase/8/docs/api/>). 2021. 08. 20. 스크린샷.
[그림 2-1] 자바 표준 라이브러리 문서

2) 파이썬의 표준 라이브러리 문서를 확인한다.



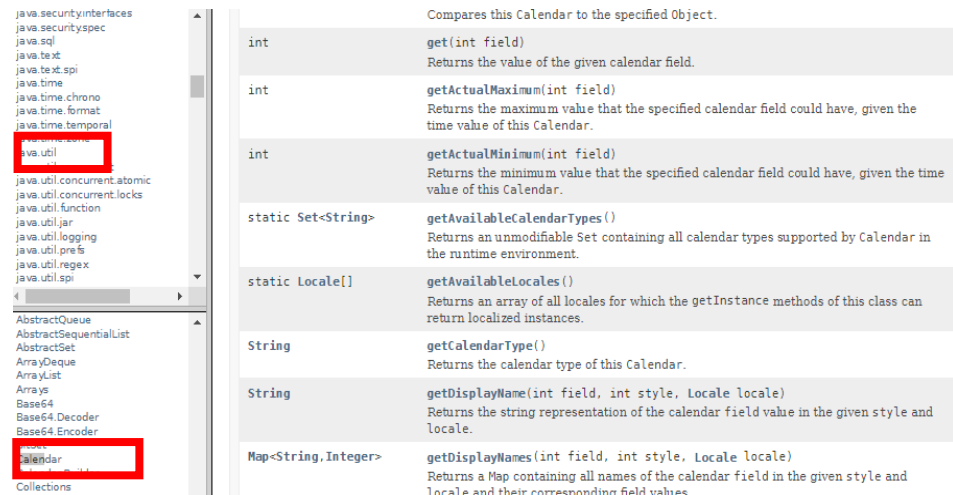
출처: 파이썬 표준 라이브러리(<https://docs.python.org/>). 2021. 07. 13. 스크린샷.
[그림 2-2] 파이썬 표준 라이브러리 문서

파이썬의 경우 <https://docs.python.org/>에 접속하여 좌측 상단의 언어와 파이썬 버전을 변경하여 라이브러리 문서 확인이 가능하다.

(나) 프로그래밍 언어의 표준 라이브러리 종류를 파악한다.

1) 자바의 표준 라이브러리 종류를 확인한다.

좌측 상단의 package를 선택한 후 좌측 하단에서 class를 선택하면 우측 창에서 Field와 Method, Description을 확인할 수 있다.



출처: 자바 표준 라이브러리 상세 내용 예시(<https://docs.oracle.com/javase/8/docs/api/>). 2021. 08. 21. 스크린샷.

[그림 2-3] 자바 표준 라이브러리 상세 항목

2) 파이썬의 표준 라이브러리 종류를 확인한다.

파이썬 라이브러리의 종류를 선택하여 애플리케이션 구현에 필요한 라이브러리의 종류를 확인한다.



출처: 파이썬 표준 라이브러리(<https://docs.python.org/ko/3/library/index.html>). 2021. 07. 13. 스크린샷.

[그림 2-4] 파이썬 표준 라이브러리 상세 항목

(다) 라이브러리의 상세 기능을 확인한다.

라이브러리에서 제공하는 기능과 사용법, 반환값을 검토하여 애플리케이션 구현에 적합한지 검토한다.

2. 해당 언어의 외부 라이브러리를 파악한다.

애플리케이션 구현에 필요한 기능이 표준 라이브러리에서 제공하지 않는 경우 인터넷 검색 등을 통해 사용 목적에 부합한 외부 라이브러리를 검색하여 적용할 수 있다. 언어별로 특정한 기능 제공을 위한 대표적인 외부 라이브러리를 파악한다.

(1) 대표적인 외부 라이브러리 종류를 파악한다.

해당 언어의 대표적인 라이브러리를 검색하고 종류 및 역할 등에 대해 확인하고 필요시 추가 검색을 진행한다.

(가) 자바의 대표적인 외부 라이브러리를 확인한다.

자바의 대표적인 외부 라이브러리에 대한 종류 및 역할에 대해 확인한 후 적용 가능 여부에 대해 검토한다.

〈표 2-4〉 자바 대표 외부 라이브러리 종류 및 역할

종류	역할
Apache Commons	자바로 구현된 라이브러리를 모은 프로젝트로 Codec, Compress, Collections, CSV, IO, Lang, Logging 등의 컴포넌트 제공
Google Guava	Apache Commons과 유사한 모듈식이며 컬렉션, I/O, 캐싱, 해싱 등의 많은 독립 라이브러리 제공
Jackson	Java Object를 JSON으로 변환하거나 JSON을 Java Object로 변환. XML/YAML/CSV 등 다양한 형식의 데이터를 지원
Log4j 2	대표적인 로깅 라이브러리로 비동기 로깅을 통한 성능 향상 및 플러그인 아키텍처, 클라우드 지원
Mockito	동작 제어가 가능한 가짜 객체(Mock)를 지원하여 테스트 프레임워크
Hibernate	Java 클래스를 데이터베이스 테이블/컬렉션과 매핑(Object Relational Mapping)

(나) 파이썬의 대표적인 외부 라이브러리를 확인한다.

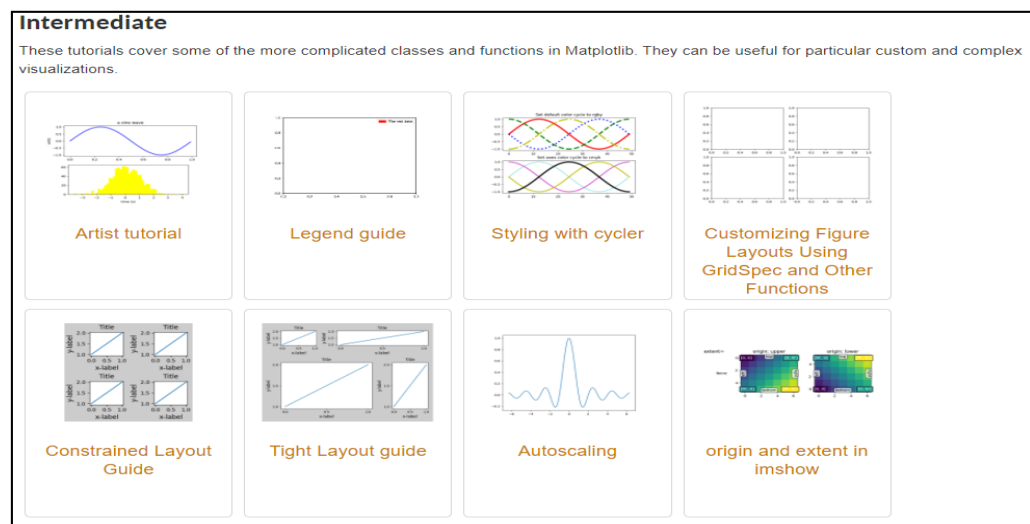
파이썬의 대표적인 외부 라이브러리에 대한 종류 및 역할에 대해 확인한 후 적용 가능 여부에 대해 검토한다.

〈표 2-5〉 파이썬 대표 외부 라이브러리 종류 및 역할

종류	역할
pandas	데이터 분석(Data Analysis) 시 데이터의 수집, 조작, 분석을 위해 사용되는 라이브러리
numpy	벡터, 행렬 등 수치 연산을 수행하는 계산 과학 분야 라이브러리
plotly	최신의 반응용, 브라우저 기반의 차트를 그릴 수 있는 시각화 라이브러리
matplotlib	데이터를 차트나 플롯(Plot)으로 그려주는 라이브러리
Scipy	분석 및 과학, 엔지니어링을 위한 여러 기본적인 작업을 위한 라이브러리
openCV	얼굴 인식 등의 영상 처리를 위한 실시간 이미지 프로세싱 오픈소스 라이브러리
fbprophet	입력받은 데이터를 기반으로 다음 값을 예측
dlib	기계 학습 및 이미지 프로세싱, 얼굴 인식 등을 위한 라이브러리

(2) 외부 라이브러리 상세 기능을 파악한다.

외부 라이브러리에서 제공하는 기능과 연동 방법, 호출/반환값 등을 검토하여 애플리케이션 구현에 적합한지 확인한다.



출처: 파이썬 Matplotlib Tutorials(<https://matplotlib.org/stable/tutorials/index.html>). 2021. 07. 13. 스크린샷.

[그림 2-5] 외부 라이브러리(Matplotlib) 제공 기능 확인 예시

③ 적용 가능한 후보 라이브러리를 대상으로 평가를 진행하고 선정한다.

1. 외부 라이브러리 선정 시 고려사항을 숙지한다.

(1) 외부 라이브러리의 사용 권한에 대해 확인한다.

오픈소스의 경우 일반적으로 라이선스와 함께 제공되므로 해당 소프트웨어를 자유롭게 사용하거나 수정할 수 있는지, 변경 사항을 재배포해야 하는지 등에 대한 확인이 필요하다.

(2) 외부 라이브러리의 지속적 관리 여부를 확인한다.

SW 구성 및 실행환경 변경, 보안 취약점 등에 정기적이고 지속적인 업데이트와 개선이 이루어지고 있는지 확인한다. 프로그래밍 언어의 버전 변경, 실행환경 변경, 신규 보안 취약점 발생 등을 지속적으로 관리하여 효율적 활용도를 높인다.

(3) 관련 프로젝트의 활성화 정도를 확인한다.

오래되고 사용빈도가 낮은 프로젝트는 시간이 지남에 따라 업데이트를 받지 못할 가능성이 존재한다. 사용빈도나 인기가 많은 프로젝트의 경우에는 일반적으로 많은 개발자가 적극적으로 사용하고 검토하여 지속적인 개선이 가능하다고 할 수 있다.

2. 외부 라이브러리의 라이선스를 파악한다.

OSS(Open Source Software)의 경우 일반적으로는 자유롭게 사용 및 복제, 배포가 가능하나, 라이선스에 따라 상업적 목적으로 이용할 수 없거나 배포 시 소스코드를 공개해야 할 수 있으니 라이선스 정책을 반드시 확인한다.



출처: 파이썬 Matplotlib 라이선스(<https://matplotlib.org/stable/users/license.html?highlight=license>). 2021. 08. 21. 스크린샷.

[그림 2-6] 파이썬 외부 라이브러리의 라이선스 확인 예시

(1) 라이선스 유형을 확인한다.

라이브러리를 제공하는 사이트의 라이선스 정책 등을 참고하여 선정한 라이브러리의 라이선스 유형을 확인한다.

〈표 2-6〉 주요 OSS 라이선스 유형

라이선스	설명
GNU GPL 2.0(GNU General Public License)	자유 소프트웨어 재단에서 만든 라이선스로 GNU 프로젝트에서 배포한 프로그램을 사용하기 위해 작성되었으며 많은 오픈소스 소프트웨어가 채택하고 있는 라이선스. 가장 많이 알려져 있고 의무사항들도 타 라이선스에 비해 엄격한 편이며, 배포하는 경우 저작권 표시, 보증책임이 없다는 표시 및 GPL에 의해 배포된다는 사실 명시 필요
GNU Lesser GPL(LGPL) 2.1	GPL 라이선스를 사용하기만 해도 소스코드를 공개해야 한다는 부담 때문에 라이브러리와 모듈로의 링크를 허용한 라이선스. 배포하는 경우, 저작권 표시, 보증책임이 없다는 표시 및 LGPL에 의해 배포된다는 사실 명시 필요
Berkeley Software Distribution(BSD) License	GPL/LGPL보다 덜 제한적이기 때문에 허용 범위가 넓으며, 가장 큰 차이점은 소스코드의 공개 의무가 없음. 수정 프로그램에 대한 소스코드의 공개를 요구하지 않기 때문에 상용 소프트웨어에 무제한 사용 가능함
Apache License	아파치 재단(ASF: Apache Software Foundation)의 모든 소프트웨어에 적용되며 BSD 라이선스와 비슷하여 소스코드 공개 등의 의무 없음. "Apache"라는 이름에 대한 상표권을 침해하지 않아야 한다는 조항이 명시적으로 포함
MIT License	미국 매사추세츠공과대학교(MIT)에서 소프트웨어 공학도들을 돕기 위해 개발한 라이선스로 소프트웨어를 누구라도 무상으로 제한 없이 취급 가능

(2) 라이선스의 정책을 확인한다.

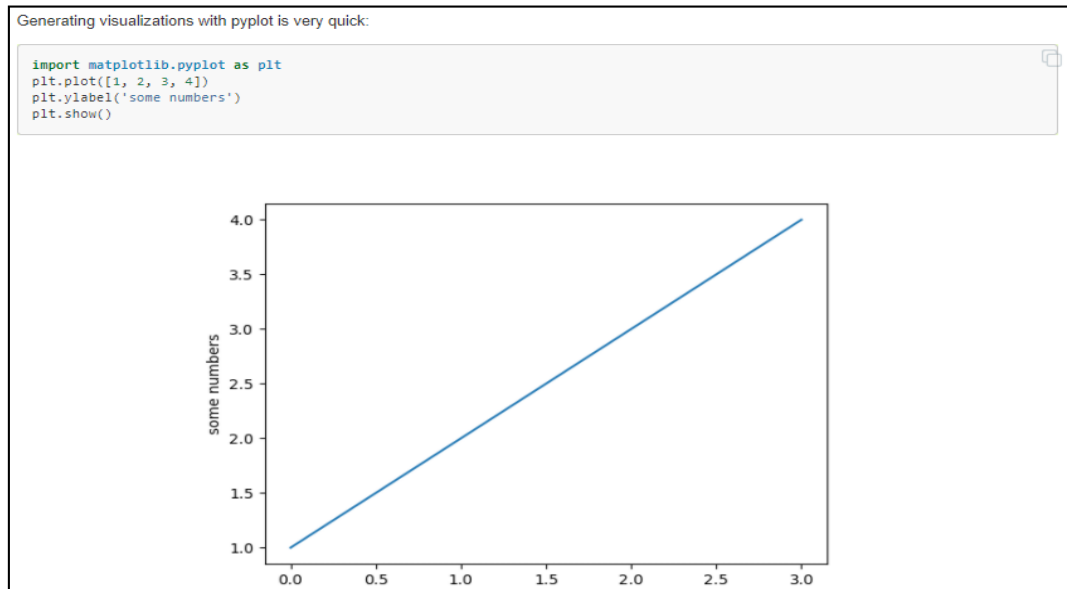
공개 SW를 이용하는 경우 공개 SW 개발자가 지정한 조건의 범위에 따라 해당 SW를 이용해야 하며, 위반하는 경우 라이선스 위반 및 저작권 침해에 따른 법적 책임을 져야 한다.

〈표 2-7〉 주요 공개 SW 라이선스별 정책 비교

라이선스	무료 이용 가능	배포 허용 가능	소스코드 수정 가능	2차적 저작물 재공개 의무	독점 SW와 결합 가능
GPL	O	O	O	O	X
LGPL	O	O	O	O	O
MPL	O	O	O	O	O
BSD License	O	O	O	X	O
Apache license	O	O	O	X	O

3. 라이브러리 적용 가이드 문서를 검토한다.

라이브러리 적용을 위한 가이드 문서의 상세 내용 및 예시 코드 등을 확인하여 애플리케이션 구현에 적용 가능한지 검토한다.



출처: Matplotlib의 pyplot 상세 안내 문서 (<https://matplotlib.org/stable/tutorials/introductory/pyplot.html#sphx-glr-tutorials-introductory-pyplot-py>). 2021. 07. 13. 스크린샷.

[그림 2-7] 외부 라이브러리(Matplotlib) 상세 기능(pyplot) 확인 예시

4. 제약 사항이 있는지 확인한다.

라이브러리를 사용하는 데 있어 프로그래밍 언어의 특성, 상호호환성, 개발환경 및 실행환경에서 발생하는 제약사항은 없는지 확인한다.

5. 라이브러리를 평가한다.

라이브러리에 대한 라이선스(이용, 배포, SW 공개 의무 등)와 제공 커뮤니티의 지원 및 활성화, 문서화 정도 등의 라이브러리 자체 영역과 요구사항 만족 여부 및 업무적 기능 구현 가능성, 시스템의 특성, 프로젝트 상황, 유지관리 효율성 등의 프로젝트 영역을 종합적으로 검토하여 후보 라이브러리들에 대한 평가를 진행한다. 평가 항목 및 방법, 점수에 대한 가중치는 프로젝트 상황을 고려하여 적용한다.

〈표 2-8〉 라이브러리 평가 항목 예시

평가 영역	평가 항목	항목 설명	점수
라이선스	활용	무료 이용 가능 및 상업적 활용 가능 여부	5
	배포	라이브러리 적용 SW에 대한 배포 가능 여부	3
	공개	소스코드 공개 의무 여부	1
커뮤니티	버전	다양한 OS 및 버전 지원, 릴리즈 주기에 대한 적합성	4
	지원	개발자의 전문성, 질의에 대한 회신 여부	5
	활성화	참여자들의 활동 현황 및 다운로드, 점유율 등	5
	문서화	설치 및 사용법 등에 대한 문서 종류와 포맷	3
개발/관리	기능성	라이브러리 적용 시 요구사항 충족 여부	5
	신뢰성	오류 발생에 대한 추적 및 복구에 대한 용이성	3
	사용성	모듈 설치 및 관리, 인터페이스 활용의 용이성	4
	성능	성능 이슈에 대한 점검 및 점검 결과 제공 여부	3
	이식성	다양한 환경에 대한 적응성 제공 여부	4
	유지관리성	주석 확인 및 소스 수정 가능 여부	5
	보안성	보안 취약점에 대한 이슈 및 점검, 패치 적용 여부	4

6. 라이브러리를 선정한다.

후보 라이브러리 중 가장 높은 점수의 라이브러리를 선정하고, 표준 라이브러리와 외부 라이브러리가 모두 존재하는 경우에는 별도의 외부 모듈 설치가 필요하지 않고 자체적으로 처리가 가능한 표준 라이브러리를 우선 적용한다.

수행 tip

- 인터넷 검색을 통해 애플리케이션 구현에 필요한 라이브러리의 존재 여부를 우선 확인한 후 라이선스의 유형과 제약 사항을 파악한다.
- 외부 라이브러리의 경우 기능과 성능, 보안 등에 대한 지속적인 모니터링 및 관리가 필요하고 자체적인 수정이 어려운 경우가 존재할 수 있어 가능한 한 표준 라이브러리 적용을 우선적으로 검토한다.

2-2. 라이브러리 구성 및 적용

학습 목표

- 애플리케이션 구현을 위해 선택한 라이브러리를 프로그래밍 언어 특성에 맞게 구성할 수 있다.
- 선택한 라이브러리를 사용하여 애플리케이션 구현에 적용할 수 있다.

필요 지식 /

① 모듈과 패키지

라이브러리는 모듈과 패키지를 모두 포함하며, 모듈의 경우 개별 파일을 지칭하고 패키지의 경우 여러 개의 파일을 모아놓은 폴더로 생각할 수 있다.

〈표 2-9〉 모듈과 패키지 비교

구분	설명	사용법
Module	한 개의 파일에서 기능을 제공한다.	import (모듈명)
Package	여러 개의 모듈을 한 개의 폴더에 묶어서 기능을 제공한다. 패키지명과 모듈명을 import하여 불러올 수 있다.	import (패키지명).(모듈명)

② PyPI(Python Package Index)

Python Package Index는 파이썬 패키지들이 모여 있는 저장소를 의미하고, Python 개발자라면 누구에게나 오픈되어 있다. PyPI에서 원하는 Python 라이브러리 설치 툴이 pip이며, 파이썬 3.4 이후의 버전에는 기본으로 포함되어 있다.

〈표 2-10〉 모듈과 패키지 비교

사용법	의미
pip list	설치된 패키지를 표시한다.
pip install <package name>	패키지를 설치한다.
pip install --upgrade <package name>	설치된 패키지를 업그레이드한다.
pip uninstall <package name>	설치된 패키지를 제거한다.
pip --help	pip와 관련된 명령어를 조회한다.

③ Maven

1. 빌드 도구(Build Tool)

Build는 소스코드 파일을 컴퓨터에서 실행할 수 있는 독립 소프트웨어 가공물로 변환하는 과정이며, Build Tool은 이러한 프로젝트의 생성과 빌드, 배포 등의 작업을 위한 도구를 의미한다.

2. Maven

아파치 메이븐(Apache Maven)은 자바용 프로젝트 관리 도구로 아파치 앤트(Ant)의 대안으로 만들어졌으며 아파치 라이선스로 배포되는 오픈소스 소프트웨어이다.

Maven은 필요한 라이브러리를 특정 문서(pom.xml)에 정의해 놓으면 내가 사용할 라이브러리뿐만 아니라 해당 라이브러리가 작동하는 데에 필요한 다른 라이브러리들까지 관리하여 네트워크를 통해서 자동으로 다운로드를 해준다.

3. 빌드 도구 비교

초기에는 Build Tool로 Ant가 많이 사용되었으나 점차 라이브러리가 많아짐에 따라 Ant를 보완한 Maven과 Gradle 등의 Build Tool로 발전하게 되었다.

〈표 2-11〉 빌드 도구 비교

구분	특징
Ant	- XML 기반의 빌드 스크립트 작성, 자유롭게 빌드 단위 지정 - 간단하고 사용하기 쉬우나 큰 프로젝트의 경우 스크립트 관리나 빌드 과정이 복잡 - 생명주기(Lifecycle)를 갖지 않아 각각의 결과물에 대한 의존관계 등을 정의
Maven	- XML 기반으로 작성되며 생명주기와 POM(Project Object Model) 개념 도입 - pom.xml 파일을 이용해 dependency를 추가하고 삭제하여 라이브러리를 관리
Gradle	- Ant와 Maven의 장점을 모아 작성되었으며 의존성 관리를 위한 다양한 방법 제공 - 빌드 스크립트를 XML이 아닌 JVM에서 실행되는 스크립트 언어인 그루비(Groovy) 사용

수행 내용 / 라이브러리 구성 및 적용하기

재료 · 자료

- 요구사항정의서 등 요구사항 관련 문서
- 프로그램 명명 규칙 및 코딩 가이드 등 개발 표준 관련 문서
- 프로그래밍 언어에 대한 라이브러리 설치 파일
- 프로그래밍 언어별 라이브러리 적용을 위한 샘플 소스코드
- 프로그래밍 언어에 대한 라이브러리 관련 문서 및 도움말

기기(장비 · 공구)

- 전산 장비: 인터넷, 컴퓨터, 프린터, 복사기, 빔 프로젝터
- 지원 도구: 문서 작성 도구, 문서 발표 도구, 소프트웨어 개발 도구, 정적 분석 도구

안전 · 유의 사항

- 소스코드 개발 및 변경 시 변경된 모듈에 대한 버전 관리에 유의한다.
- 실습 환경 및 프로그래밍 언어, 개발 툴에 따른 라이브러리 설치 방법이 달라질 수 있으니 인터넷 검색을 통해 추가 확인하도록 한다.
- 검증되지 않은 사이트에서 파일을 다운받아 설치하는 경우 보안적인 위험이 발생할 수 있으므로 공식 사이트에서 다운받도록 한다.
- 라이브러리 적용 후 요구사항과 애플리케이션 기능에 적합한지 검토하고, 다른 프로그램에 영향을 미칠 수 있으므로 단위 및 통합, 회귀 테스트 등을 통해 기능 점검을 진행한다.

수행 순서

① 선정한 라이브러리 적용 방법에 대해 파악한다.

1. 프로그래밍 언어의 구성이 어떻게 이루어지는지 파악한다.

프로그래밍 언어에 따라 라이브러리를 구성하기 위한 프로그램의 구조와 구성 방법을 파악한다.

(1) 프로그래밍 언어별 구조를 파악한다.

객체지향 프로그래밍 언어의 경우 객체, 클래스, 메소드의 개념을 통해 프로그램

을 구현하므로, 파이썬이나 자바로 프로그래밍을 하는 경우 객체지향 프로그래밍의 구조적인 특징을 고려하여 설치 및 호출, 연동 방법을 확인한다.

(2) 라이브러리 구성 방법을 파악한다.

라이브러리 구성을 위해서는 명령어를 이용하여 라이브러리를 불러와 사용하며 파이썬을 비롯한 일반적인 경우 import 명령어가 사용된다.

2. 프로그래밍 언어 특성에 맞는 구성 방법을 파악한다.

라이브러리를 제공하는 사이트에서는 라이브러리의 제공 기능과 기능 구현을 위한 예시 소스코드, 가이드 문서 등을 제공하고 있으므로 해당 자료를 확인한다.

(1) 라이브러리 적용을 위한 도움말을 확인한다.

라이브러리를 다운받아 설치하거나 호출하는 방법 등에 관한 문서 및 도움말을 확인한다.

(2) 라이브러리를 적용한 예시 소스를 확인한다.

프로그래밍 언어에서 라이브러리를 호출하고 연동하는 예시 프로그램 소스를 확인하거나 인터넷 검색을 통해 적용 방법을 확인한다.

② 표준 라이브러리를 애플리케이션 구현에 적용한다.

1. 표준 라이브러리 적용 방법을 파악한다.

(1) 파이썬의 표준 라이브러리 적용 방법을 파악한다.

한 모듈에 있는 파이썬 코드는 임포팅이라는 프로세스를 통해 다른 모듈에 있는 코드들에 대한 접근권을 획득할 수 있으며, import문 사용 방법에 따라 애플리케이션에서 라이브러리 호출 및 사용을 위한 소스코드가 달라질 수 있다.

〈표 2-12〉 파이썬 표준 라이브러리 적용 및 사용 코드 예시

적용 방법	의미	사용 코드
import os	os 모듈 전체를 불러와 사용	os.getcwd()
from os import *	os 모듈 전체를 불러오는 것은 동일하나 코드는 변수/함수만 사용	getcwd()
from os import getcwd	os 모듈 내의 특정 함수(getcwd)만 불러와 사용	getcwd()

(2) 자바의 표준 라이브러리 적용 방법을 확인한다.

자바의 경우에도 import문을 통해 표준 라이브러리를 연동하여 소스코드에서 호출 및 사용이 가능하다.

〈표 2-13〉 자바 표준 라이브러리 적용 및 사용 코드 예시

적용 방법	의미	사용 코드
import java.util.Date	Date 클래스만 불러와 사용	Date
import java.util.*	해당 패키지의 클래스 전체를 불러와 사용	Date

2. 표준 라이브러리를 애플리케이션에 적용한다.

애플리케이션 구현에 필요한 표준 라이브러리를 프로그래밍 언어에 맞게 적용한다.

〈표 2-14〉 표준 라이브러리 적용 예시

적용 방법	파이썬	자바
현재 날짜 조회	<pre>from datetime import datetime print(datetime.today())</pre>	<pre>import java.util.Date; public class Exam1 { public static void main(String[] args) { Date now = new Date(); System.out.println(now); } }</pre>
난수 생성	<pre>import random print(random.random())</pre>	<pre>import java.util.Random; public class Exam2 { public static void main(String[] args) { Random ran = new Random(); System.out.println(ran.nextInt()); } }</pre>

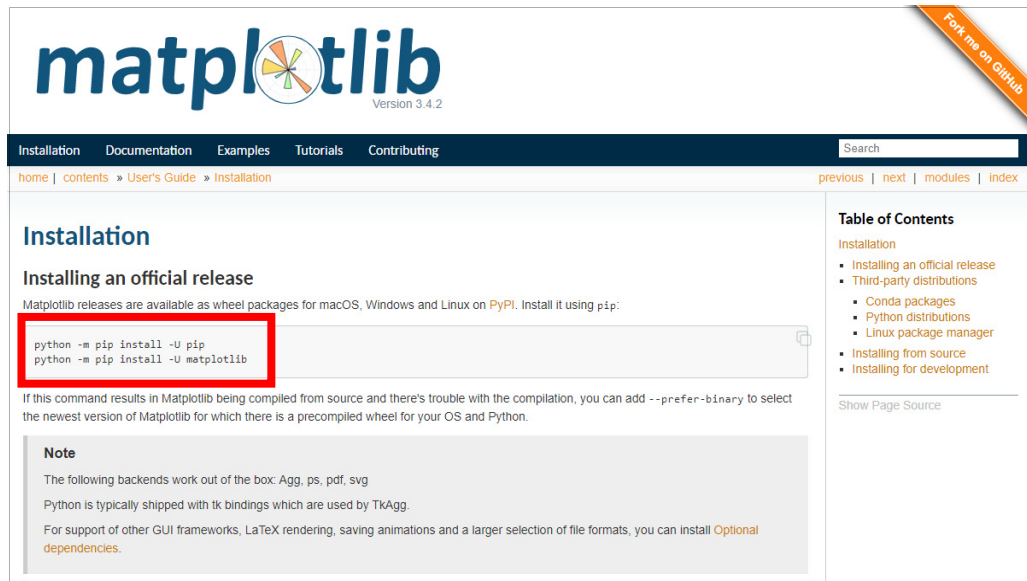
③ 외부 라이브러리를 애플리케이션 구현에 적용한다.

1. 외부 라이브러리 설치 방법을 확인한다.

애플리케이션 구현에 필요한 외부 라이브러리를 제공하는 공식 사이트 및 인터넷 검색 등을 통해 설치 및 사용 방법을 확인한다.

(1) 파이썬의 외부 라이브러리 적용 방법을 확인한다.

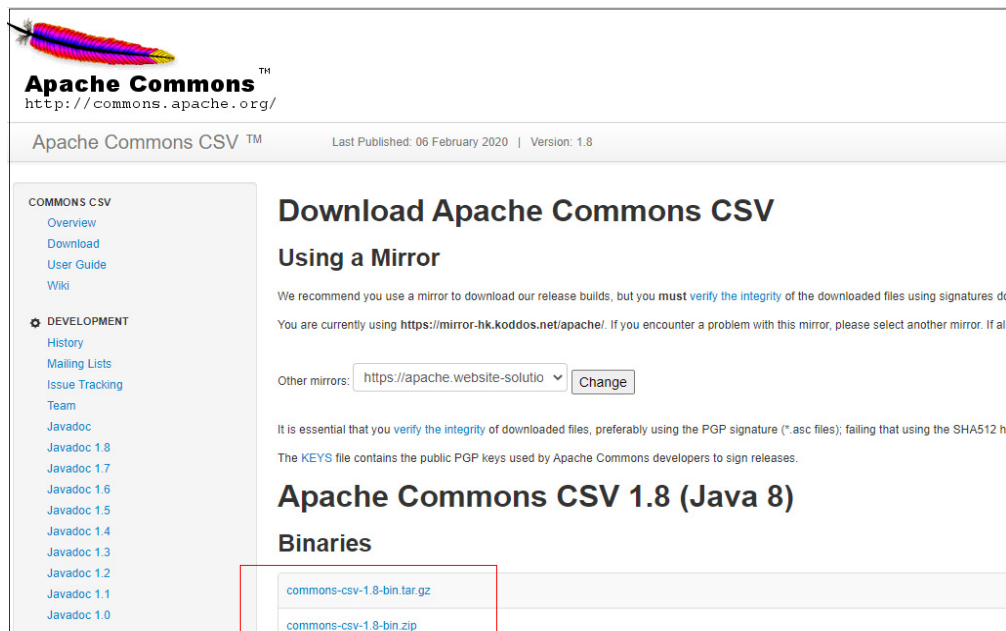
파이썬의 경우 일반적으로 pip를 이용하여 외부 라이브러리를 설치할 수 있으며, 관련 공식 사이트에서 제공하는 Installation 방법을 참고하여 설치할 수 있다.



출처: matplotlib install 안내 화면(<https://matplotlib.org/stable/users/installing.html>). 2021. 07. 13. 스크린샷.
[그림 2-8] 파이썬 외부 라이브러리 설치 안내 화면 예시

(2) 자바의 외부 라이브러리 적용 방법을 확인한다.

자바의 경우 해당 사이트에서 라이브러리를 직접 다운받은 후 개발 도구에서 라이브러리를 추가하여 사용할 수 있다.



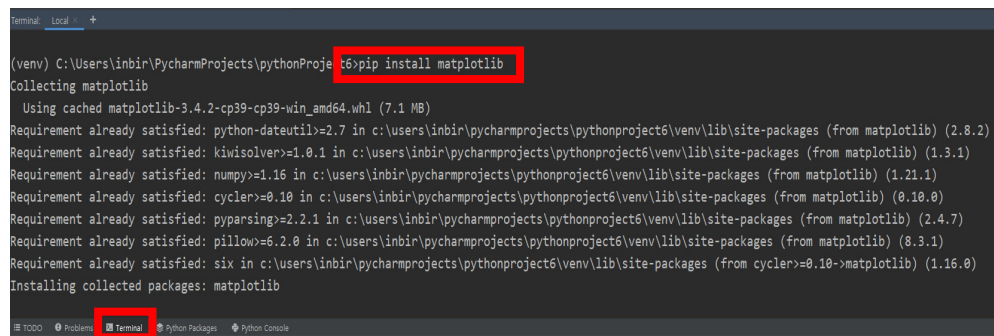
출처: Apache Commons CSV 라이브러리 설치 안내 화면(http://commons.apache.org/proper/commons-csv/download_csv.cgi). 2021. 07. 30. 스크린샷.
[그림 2-9] 자바 외부 라이브러리 설치 안내 화면 예시

2. 외부 라이브러리를 설치한다.

(1) 파이썬 외부 라이브러리를 설치한다.

(가) pip 명령을 이용하여 외부 라이브러리를 설치한다.

CMD 명령어 창이나 Pycharm 하단의 Terminal 탭을 선택한 후 표시되는 화면에서 pip를 이용하여 외부 라이브러리를 설치한다.

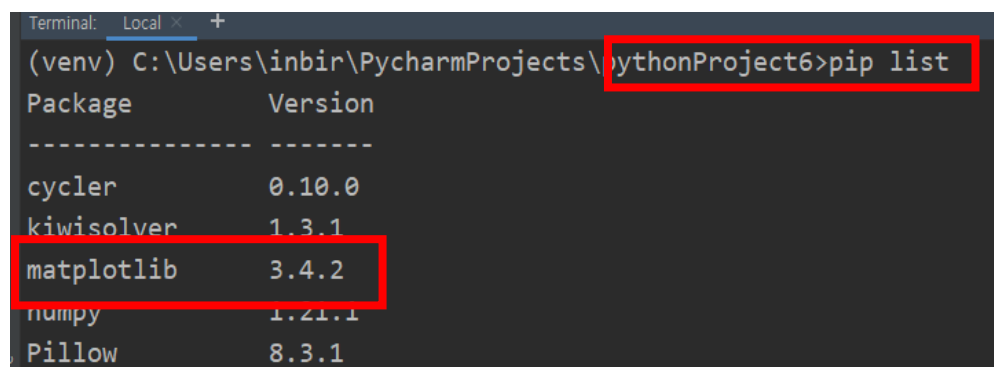


```
(venv) C:\Users\inbir\PycharmProjects\pythonProject6>pip install matplotlib
Collecting matplotlib
  Using cached matplotlib-3.4.2-cp39-cp39-win_amd64.whl (7.1 MB)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\inbir\pycharmprojects\pythonproject6\venv\lib\site-packages (from matplotlib) (2.8.2)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\inbir\pycharmprojects\pythonproject6\venv\lib\site-packages (from matplotlib) (1.3.1)
Requirement already satisfied: numpy>=1.16 in c:\users\inbir\pycharmprojects\pythonproject6\venv\lib\site-packages (from matplotlib) (1.21.1)
Requirement already satisfied: cycler>=0.10 in c:\users\inbir\pycharmprojects\pythonproject6\venv\lib\site-packages (from matplotlib) (0.10.0)
Requirement already satisfied: pyparsing>=2.2.1 in c:\users\inbir\pycharmprojects\pythonproject6\venv\lib\site-packages (from matplotlib) (2.4.7)
Requirement already satisfied: pillow>=6.2.0 in c:\users\inbir\pycharmprojects\pythonproject6\venv\lib\site-packages (from matplotlib) (8.3.1)
Requirement already satisfied: six in c:\users\inbir\pycharmprojects\pythonproject6\venv\lib\site-packages (from cycler>=0.10->matplotlib) (1.16.0)
Installing collected packages: matplotlib
```

출처: 집필진 제작(2021)
[그림 2-10] 파이썬 외부 라이브러리 설치 화면 예시

(나) 정상 설치 여부를 확인한다.

해당 라이브러리가 정상적으로 설치되었는지 확인하기 위하여 pip list를 활용하여 설치된 패키지 목록을 조회한다.



```
(venv) C:\Users\inbir\PycharmProjects\pythonProject6>pip list
Package            Version
-----
cycler              0.10.0
kiwisolver          1.3.1
matplotlib          3.4.2
numpy               1.21.1
Pillow              8.3.1
```

출처: 집필진 제작(2021)
[그림 2-11] 파이썬 설치 라이브러리 및 버전 확인 예시

(2) 자바 외부 라이브러리를 설치한다.

관련 커뮤니티 및 제공 사이트에서 개발자 환경에 적합한 라이브러리 파일을 직접 다운받거나 Maven의 dependency 추가를 통해 설치할 수 있으며, 자체 개발한 라이브러리를 설치할 수 있다.

(가) 외부 라이브러리를 다운받아 직접 설치한다.

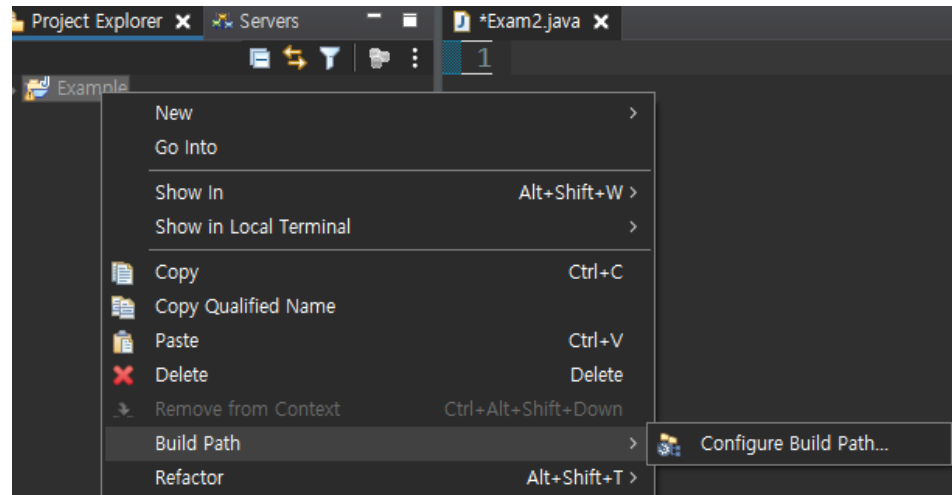
1) 관련 사이트에서 라이브러리 파일을 다운받는다.

다운받은 압축 파일을 특정 디렉터리에 압축 해제하여 jar 파일이 정상적

으로 포함되어 있는지 확인한다.

2) 이클립스에서 외부 라이브러리 추가를 선택한다.

프로젝트를 선택한 후 우클릭을 누르면 표시되는 메뉴에서 Build Path > Configure Build Path..를 선택한다.

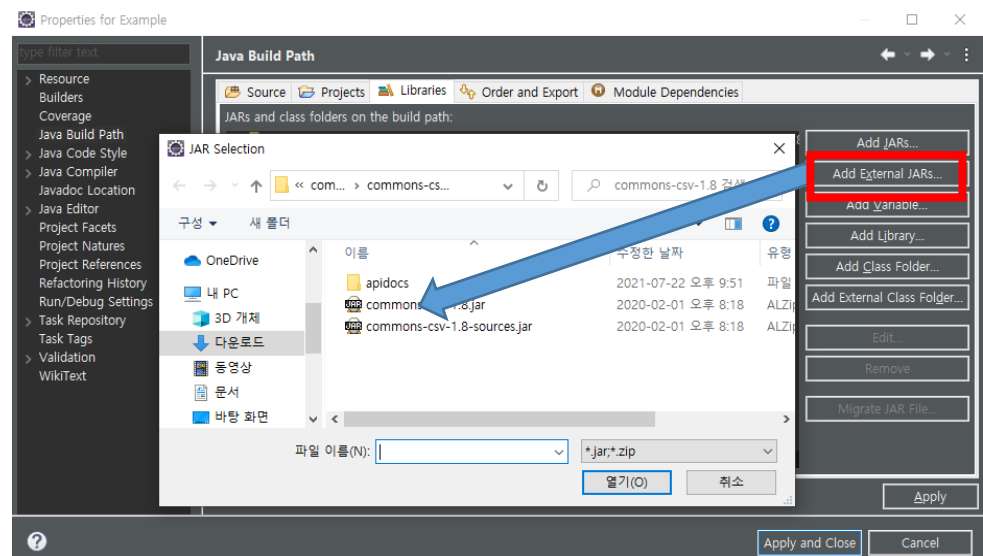


출처: 집필진 제작(2021)

[그림 2-12] 이클립스 외부 라이브러리 설치 메뉴

3) jar 파일을 라이브러리에 추가한다.

우측의 'Add External JARs..' 메뉴를 선택한 후 파일을 선택할 수 있는 창에서 압축 해제한 JAR 파일을 선택한다. 하단의 'Apply and Close' 버튼을 선택하여 설치를 완료한다.



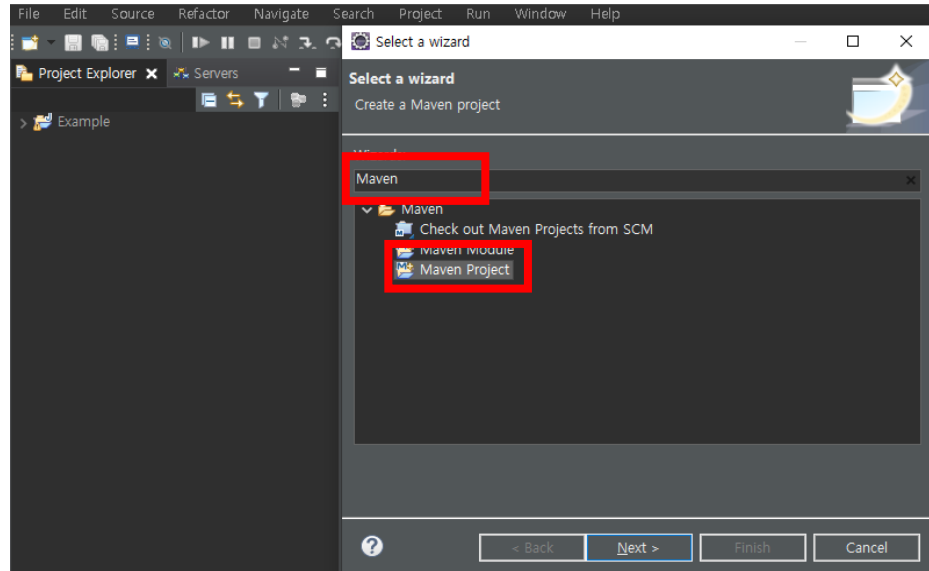
출처: 집필진 제작(2021)

[그림 2-13] 이클립스 외부 라이브러리 선택 및 설치

(나) Maven을 이용하여 외부 라이브러리를 설치한다.

1) 이클립스에서 신규 프로젝트를 생성한다.

File > New > Project.. 메뉴를 선택한 후 표시되는 새 창에서 Maven으로 검색한 후 'Maven Project'를 선택한다.

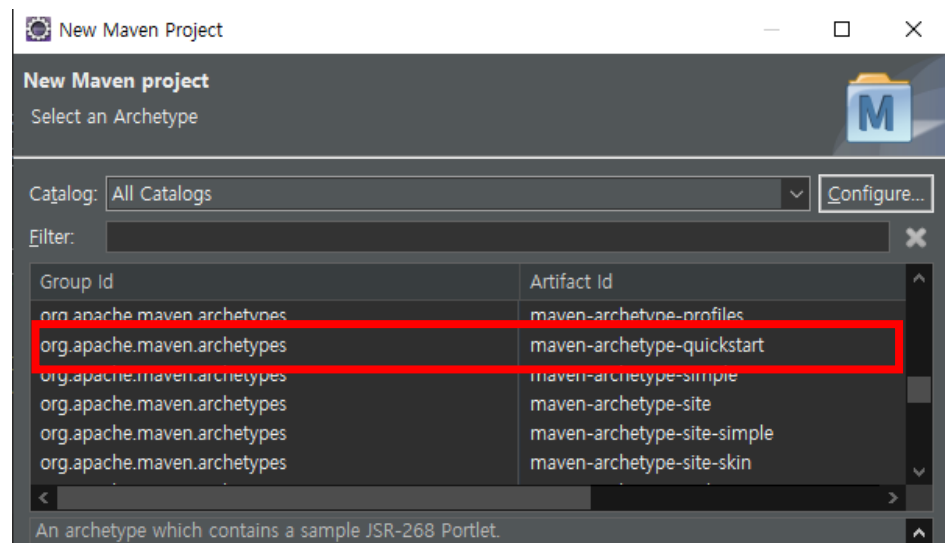


출처: 집필진 제작(2021)

[그림 2-14] 이클립스 Maven 프로젝트 신규 생성

2) Maven 프로젝트 생성을 위한 기본 설정값을 지정한다.

Archetype 선택 화면에서 'maven-archetype-quickstart'를 선택한 후 Group ID(프로젝트 구조)와 Artifact ID(프로젝트명)를 입력하여 기본 설정값을 저장한다.



출처: 집필진 제작(2021)

[그림 2-15] 이클립스 Maven 프로젝트 기본 설정 선택

3) 해당 사이트에서 Maven 설정을 위한 방법을 확인한다.

Maven Repository에서 dependency 추가를 위한 groupId, artifactId, version 정보를 확인한다.



출처: Apache Commons CSV 라이브러리 Maven Repository 설치 안내 화면(<https://commons.apache.org/proper/commons-csv/>. 2021. 07. 30. 스크린샷.

[그림 2-16] 자바 외부 라이브러리 Maven 설치 안내 화면 예시

4) dependency를 추가한다.

이클립스 Maven 프로젝트의 pom.xml 파일을 선택한 후 dependency를 추가한다.

〈표 2-15〉 자바 표준 라이브러리 적용 및 사용 코드 예시

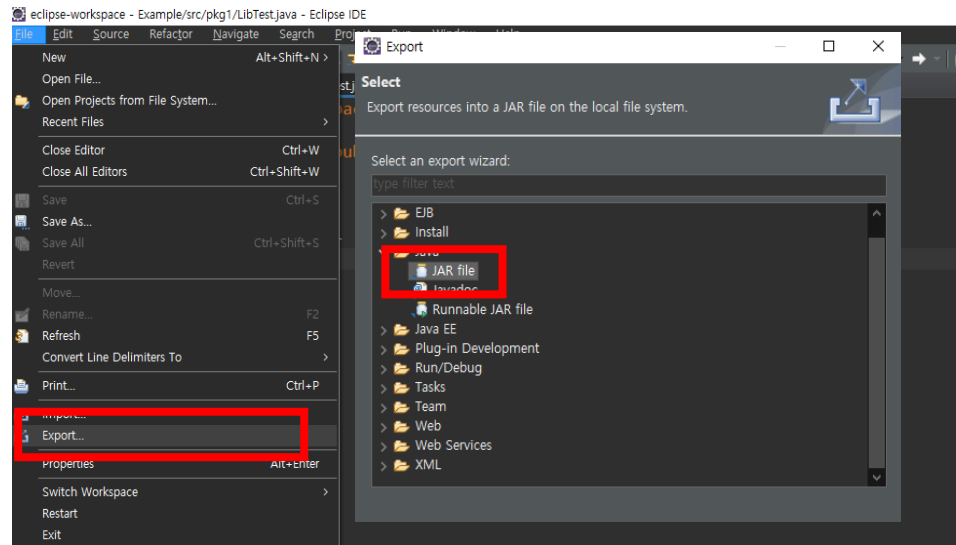
파일명	dependency 추가
pom.xml	<dependencies>
	...
	<dependency>
	<groupId>org.apache.commons</groupId>
	<artifactId>commons-csv</artifactId>
	<version>1.8</version>
	</dependency>
	...
	</dependencies>

(다) 자체 개발한 라이브러리를 jar로 배포하여 설치한다.

비즈니스 특성 등으로 인해 기능 구현을 위한 표준/외부 라이브러리가 존재하지 않는 경우 이미 개발한 기능을 라이브러리 형태로 배포하여 타 시스템에서 활용 가능하다.

1) Export를 선택한다.

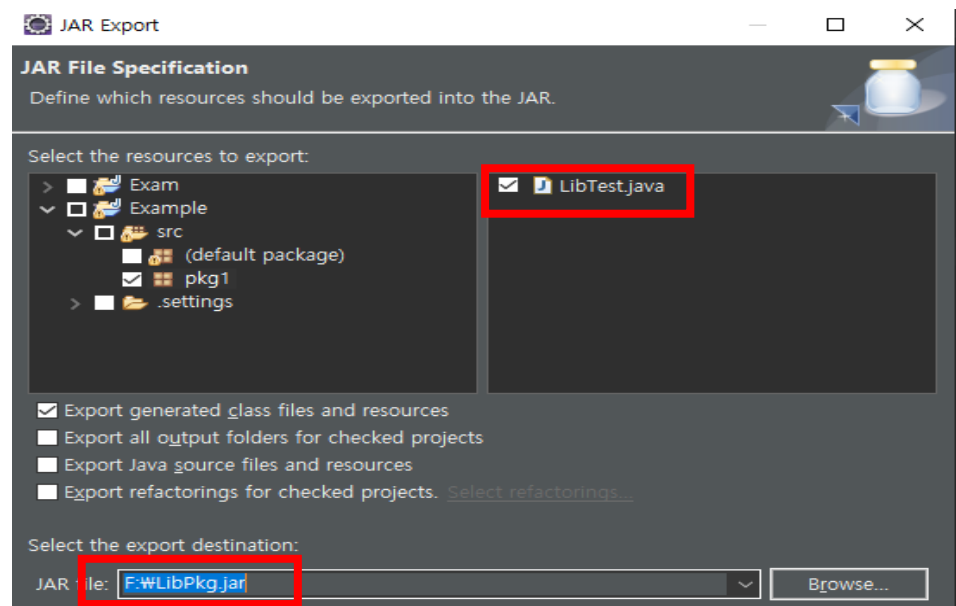
File > Export 메뉴를 선택한 후 표시되는 새 창에서 JAVA > JAR file을 선택한다.



출처: 집필진 제작(2021)
[그림 2-17] 이클립스 Export

2) 배포 대상 및 저장 파일명을 선택한다.

라이브러리 형태로 배포가 필요한 프로젝트 혹은 모듈을 선택한 후 저장할 JAR file을 입력한다.



출처: 집필진 제작(2021)
[그림 2-18] 이클립스 배포 대상 선택 및 저장 파일명 지정

3) 저장된 JAR 파일을 확인한다.

저장된 JAR 파일을 압축 해제하여 패키지 구조와 동일하게 디렉터리가 있고 .class 파일이 존재하는지 확인한다.

4) JAR 파일을 배포한다.

JAR 파일을 전달하고 활용이 필요한 시스템의 경우 외부 라이브러리를 설치하는 방법과 동일한 방법으로 설치한다.

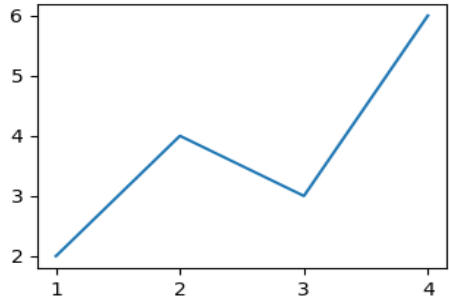
3. 외부 라이브러리를 애플리케이션에 적용한다.

파이썬과 자바 모두 import문을 통해 외부 라이브러리를 불러와 사용할 수 있으며 공식 사이트에서 제공하는 샘플 코드 및 인터넷 검색 등을 통해 애플리케이션에 적용한다.

(1) 파이썬의 외부 라이브러리를 애플리케이션에 적용한다.

import문으로 설치된 외부 라이브러리를 적용할 수 있으며, matplotlib의 경우 <https://matplotlib.org/stable/tutorials/index.html>에서 여러 가지 샘플 코드 작성법 확인이 가능하다.

〈표 2-16〉 파이썬 외부 라이브러리 적용 및 사용 코드 예시

작성 코드 예시	실행 결과
<pre>import matplotlib.pyplot as plt plt.plot([1, 2, 3, 4], [2, 4, 3, 6]) plt.show()</pre>	 <p>출처: 집필진 제작(2021)</p>

(2) 자바의 외부 라이브러리를 애플리케이션에 적용한다.

파이썬과 동일하게 import문을 사용하여 설치한 외부 라이브러리를 호출하여 애플리케이션 구현이 가능하다. 서로 다른 프로그래밍 언어로 작성된 라이브러리 적용도 가능하며, 파이썬으로 작성한 자체 모듈을 자바 프로그램에서 호출하는 방법에 대해서도 확인한다.

(가) 자바로 된 라이브러리를 애플리케이션에 적용한다.

설치한 자바 외부 라이브러리 적용을 위하여 import문으로 라이브러리를 호출하고, 라이브러리에서 제공하는 API를 참고하여 애플리케이션 기능을 구현한다.

〈표 2-17〉 자바 외부 라이브러리 적용 및 사용 코드 예시

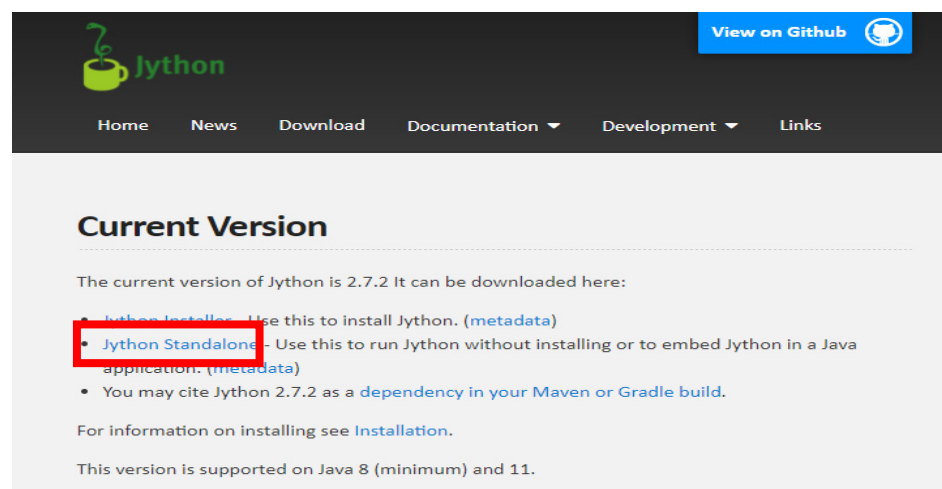
작성 코드 예시	CSV 파일 내용	실행 결과
<pre>import java.io.FileReader; import java.io.Reader; import org.apache.commons.csv.CSVFormat; import org.apache.commons.csv.CSVRecord; public class Exam2 { public static void main(String[] args) throws Exception { Reader in = new FileReader("F:file.csv"); Iterable<CSVRecord> records = CSVFormat.EXCEL.parse(in); for (CSVRecord record: records) { System.out.println(record.get(0) + "/" + record.get(1) + "/" + record.get(2)); } } }</pre>	<p>F:file.csv 파일</p> <pre>Name,Age,Weight GilDong,18,60 GapDong,16,64</pre>	<p>Name/Age/Weight</p> <pre>GilDong/18/60 GapDong/16/64</pre>

(나) 파이썬으로 작성한 라이브러리를 애플리케이션에 적용한다.

파이썬으로 작성한 라이브러리 연동을 위하여 Jython을 설치한 후 파이썬 라이브러리를 적용한다.

1) Jython을 다운받아 외부 라이브러리에 등록한다.

자이썬(Jython)은 자바 플랫폼용 파이썬으로 관련 사이트에서 JAR파일을 다운로드받은 후 외부 라이브러리 설치 방법(Build Path > Configure Build Path..)과 동일한 방식으로 설치한다.



출처: Jython 다운로드 화면(<https://www.jython.org/download>). 2021. 08. 10. 스크린샷.
[그림 2-19] 자바에서 파이썬 연동을 위한 Jython 다운로드 화면

2) 간단한 파이썬 모듈을 작성한다.

연동 테스트를 위하여 간단한 파이썬(.py) 모듈을 작성한다.

〈표 2-18〉 자바 연동을 위한 두 수의 덧셈과 뺄셈 연산 파일 예시

파이썬 파일명	소스 예시
add_minus.py	def add(a, b) :
	return a + b
	def minus(a, b) :
	return a - b

3) Jython을 연동하여 파이썬 모듈 및 함수를 호출한다.

Jython을 import하여 상단에서 작성한 파이썬 파일명과 함수를 호출하여 정상적으로 결과가 표시되는지 확인한다.

〈표 2-19〉 자바 연동을 위한 두 수의 덧셈과 뺄셈 연산 파일

소스 예시	실행 결과
import org.python.util.PythonInterpreter;	
public class PyExam {	
private static PythonInterpreter interpreter;	
public static void main(String[] args) {	
interpreter = new PythonInterpreter();	25
interpreter.execfile("add_minus.py");	
interpreter.exec("print(add(10,15))");	
}	
}	

④ 라이브러리 적용에 따른 기능 점검을 수행하고 관리 체계를 구성한다.

라이브러리가 적용된 후에는 테스트를 수행하여 기능에 문제가 없는지 확인하고, 테스트를 통해 점검한다.

1. 라이브러리 적용에 따른 기능 점검을 수행한다.

(1) 컴파일 및 실행이 정상적으로 되는지 확인한다.

라이브러리 파일이 정상적으로 설치되지 않거나 import가 되지 않은 경우 해당 패키지나 모듈을 찾을 수 없다는 오류가 발생하므로, 이런 경우 라이브러리 파일의 설치 및 설정을 다시 점검하여 수정한다.

(2) 라이브러리 제공 기능이 애플리케이션 기능 구현에 적합한지 확인한다.

라이브러리를 통해 애플리케이션에서 필요한 기능이 정상적으로 동작하는지 확인한다. 최신 라이브러리는 발견되지 않은 오류를 가지고 있을 수 있으므로 안정된 라이브러리 버전을 이용한다.

(3) 단위 및 통합 테스트를 수행한다.

라이브러리 적용 기능에 대한 단위 테스트 케이스를 식별하여 테스트를 수행하고 단위 테스트가 정상적으로 종료되면, 시스템 안정성 확인을 위한 통합 테스트를 진행한다.

2. 외부 라이브러리에 대한 관리 체계를 구성한다.

사용하고 있는 외부 라이브러리를 지속적으로 추적하고 관리할 수 있는 체계는 시스템의 개발 및 유지관리 측면에서 상당히 중요하다.

(1) 외부 라이브러리 사용 현황 및 사용 목적을 파악한다.

프로젝트 및 시스템에서 사용하고 있는 외부 라이브러리의 사용 용도를 확인한다. 유사한 목적으로 사용되는 외부 라이브러리의 경우 동일한 외부 라이브러리로 통일하거나 표준 라이브러리로 대체, 혹은 별도의 상용 솔루션 구매 등을 위한 기초 자료로 활용될 수 있다.

(2) 외부 라이브러리의 라이선스 및 수정 여부를 확인한다.

관련 커뮤니티 등을 확인하여 해당 라이브러리의 라이선스 정책 등을 검토하고, 라이브러리에 대한 단순 사용인지 수정 사용인지 등에 대해 확인한다.

(3) 외부 라이브러리 사용 목록을 작성한다.

외부 라이브러리 관리를 위한 담당자를 지정하고, 파악된 사용 현황을 기준으로 관리 목록을 작성한다.

〈표 2-20〉 외부 라이브러리 사용 목록 작성 예시

라이브러리	유형	라이선스	수정 여부	사용 용도	담당자
commons-fileupload-1.2.1	jar	Apache1.1	X	파일 업로드	김갑○
commons-lang-2.0	jar	Apache2.0	X	유틸리티	김을○
CKEditor(4.5.6)	JS	GPL,LGPL	X	공지사항 관리	김병○
Mwphotobrowser	java	MIT	O	열람 뷰어	김정○

(4) 외부 라이브러리에 대한 취약점 등을 주기적으로 점검한다.

사용하고 있는 외부 라이브러리에 대한 보안 취약점과 성능, 기능 개선 등을 위한 패치 여부를 주기적으로 점검하여 적용한다.

공개SW 소개

공개SW 사업

정보마당

열린마당

Open UP

정보마당

공개SW 활용 가이드

공개SW 보안취약점 >

공개SW 활용 성공사례

공개SW 기술지원

공개SW R&D허브

자료실

과학기술정보통신부

Ministry of Science and ICT

nipa

정보통신산업진흥원

National IT Industry Promotion Agency

KOPF

한국공개SW

활성화포럼

Home > 정보마당 > 공개SW 보안취약점

공개SW 보안취약점

분류

정렬

번호	컴포넌트 명 및 버전	취약점ID	심각도	취약점 최종 보고일	대응방안
6	Spring Data Commons 1.8.4.RELEASE	CVE-2018-1273	7.5 (High)	2019/10/10	
5	Spring Data Commons 1.6.1.RELEASE	CVE-2018-1273	7.5 (High)	2019/10/10	
4	Apache Commons Collections 2.1	CVE-2015-6420	7.5 (High)	2018/10/02	
3	Spring Data Commons 1.10.0.RELEASE	CVE-2018-1273	7.5 (High)	2019/10/10	
2	Apache Commons FileUpload 1.3.1	CVE-2016-1000031	9.8 (Critical)	2016/10/25	
1	Apache Commons Collections 3.2.1	CVE-2017-15708	9.8 (Critical)	2018/01/08	

컴포넌트명

commons

검색

출처: 공개 SW 보안 취약점 확인 화면(https://www.oss.kr/info_sec?search_target=component_name&search_keyword=commons). 2021. 07. 30. 스크린샷.

[그림 2-20] 공개 SW에 대한 보안 취약점 점검 예시

수행 tip

- 표준/외부 라이브러리에는 없으나 특정 업무 처리를 위한 공통 모듈이 필요한 경우 자체 개발하여 라이브러리 형태로 배포 및 활용이 가능하다.
- 공개 SW 및 외부 라이브러리의 경우 성능, 보안 등의 이슈 등으로 패치가 필요할 수 있어 사용 현황과 관련 시스템에 대한 체계적인 관리가 필요하다.

교수 방법

- 라이브러리의 개념과 역할, 종류에 대해 설명한다.
- 프로그래밍 언어별 표준 라이브러리와 외부 라이브러리의 차이와 제공 기능에 대해 설명한다.
- 라이브러리에 대한 검색 방법 및 선택 시 유의사항에 대해 설명한다.
- 프로그래밍 언어 특성에 맞게 라이브러리를 적용하는 방법을 설명한다.
- 라이브러리 적용 후 단위/통합/회귀 테스트를 통해 검증하는 방법을 설명한다.

학습 방법

- 라이브러리에 대한 개념 및 역할에 대해 파악한다.
- 프로그래밍 언어에 맞는 라이브러리의 종류를 파악한다.
- 기능 구현에 필요한 라이브러리를 검색하고 선정하는 방법을 실습한다.
- 표준 라이브러리 및 외부 라이브러리를 언어에 맞게 설치하고 적용 방법을 실습한다.
- 라이브러리 적용 후 테스트를 통해 검증하는 절차를 실습한다.

평가 준거

- 평가자는 학습자가 학습 목표를 성공적으로 달성하였는지를 평가해야 한다.
- 평가자는 다음 사항을 평가해야 한다.

학습 내용	학습 목표	성취수준		
		상	중	하
라이브러리 선정	- 애플리케이션에 필요한 라이브러리를 선정할 수 있다.			
라이브러리 구성 및 적용	- 애플리케이션 구현을 위해 선택한 라이브러리를 프로그래밍 언어 특성에 맞게 구성할 수 있다.			
	- 선택한 라이브러리를 사용하여 애플리케이션 구현에 적용할 수 있다.			

평가 방법

- 문제해결 시나리오

학습 내용	평가 항목	성취수준		
		상	중	하
라이브러리 선정	- 기능 구현에 필요한 표준 및 외부 라이브러리를 검색할 수 있는 능력			
	- 애플리케이션 특성을 고려하여 라이브러리를 선정할 수 있는 능력			
라이브러리 구성 및 적용	- 애플리케이션 구현을 위해 선택한 라이브러리를 프로그래밍 언어 특성에 맞게 적용할 수 있는 능력			
	- 선택한 라이브러리를 사용하여 올바르게 구현되었는지 여부를 검증할 수 있는 능력			

• 평가자 체크리스트

학습 내용	평가 항목	성취수준		
		상	중	하
라이브러리 선정	- 라이브러리의 유형과 종류, 표준/외부 라이브러리의 특징에 대한 숙지 여부			
	- 라이브러리 선정 시 우선순위 및 고려사항, 제약사항에 대한 숙지 여부			
라이브러리 구성 및 적용	- 표준 라이브러리와 외부 라이브러리를 프로그래밍 언어 특성에 맞게 구성할 수 있는 능력			
	- 선정한 라이브러리를 이용하여 애플리케이션을 구현할 수 있는 능력			

피드백

1. 문제해결 시나리오

- 애플리케이션 구현에 필요한 표준 라이브러리와 외부 라이브러리를 검색할 수 있는지 평가하고 올바른 검색 방법과 수정이 필요한 내용을 알려준다.
- 요구사항 및 애플리케이션 특성을 고려하여 라이브러리를 선정할 수 있는지 평가하고 수정이 필요한 내용을 알려 준다.
- 표준 라이브러리와 외부 라이브러리를 애플리케이션에 올바르게 적용하였는지 평가하고 오류가 발생하는 경우 해결 방법을 알려준다.
- 문제해결 시나리오 결과가 '상'인 경우 다른 학습자와 모범 사례를 공유하고 설명할 수 있도록 안내한다.
- 문제해결 시나리오 결과가 '중'인 경우 부족한 부분을 보완할 수 있도록 설명한다.
- 문제해결 시나리오 결과가 '하'인 경우 모범 사례를 참조하여 다시 학습할 수 있도록 유도한다.

2. 평가자 체크리스트

- 라이브러리에 대한 개념과 유형, 종류, 차이점을 숙지하고 있는지 평가하고 잘못된 내용이 있는 경우 수정이 필요한 부분을 알려준다.
- 라이브러리를 선정하고 적용하는 경우 고려가 필요한 사항에 대해 학습자 간에 경험이나 사례를 공유할 수 있도록 한다.
- 평가자 체크리스트 결과가 '상'인 경우 다른 학습자와 모범 사례를 공유하고 설명할 수 있도록 안내한다.
- 평가자 체크리스트 결과가 '중'인 경우 부족한 부분을 보완할 수 있도록 설명한다.
- 평가자 체크리스트 결과가 '하'인 경우 모범 사례를 참조하여 다시 학습할 수 있도록 유도한다.



- 공개 SW에 대한 보안 취약점 점검. https://www.oss.kr/info_sec?search_target=component_name&search_keyword=commons에서 2021. 07. 30. 검색.
- 외부 라이브러리(Matplotlib) 제공 기능 확인. <https://matplotlib.org/stable/tutorials/index.html>에서 2021. 07. 13. 검색.
- 자바 개발 툴(Eclipse) 설치 파일 다운로드. <https://www.eclipse.org/>에서 2021. 07. 10. 검색.
- 자바 설치 파일 다운로드. <https://www.oracle.com/java/technologies/javase-downloads.html>에서 2021. 07. 10. 검색.
- 자바에서 파이썬 연동을 위한 Jython 다운로드 화면. <https://www.jython.org/download>에서 2021. 08. 10. 검색.
- 자바 외부 라이브러리 설치 안내 화면. http://commons.apache.org/proper/commons-csv/download_csv.cgi에서 2021. 07. 30. 검색.
- 자바 표준 라이브러리 문서. <https://docs.oracle.com/javase/8/docs/api/>에서 2021. 08. 20. 검색.
- 자바 표준 라이브러리 상세 항목. <https://docs.oracle.com/javase/8/docs/api/>에서 2021. 08. 21. 검색.
- 파이썬 설치 파일 다운로드. <https://www.python.org/downloads/>에서 2021. 07. 10. 검색.
- 파이썬 외부 라이브러리 설치 안내 화면. <https://matplotlib.org/stable/users/installing.html>에서 2021. 07. 13. 검색.
- 파이썬 외부 라이브러리의 라이선스 확인. <https://matplotlib.org/stable/users/license.html?highlight=license>에서 2021. 08. 21. 검색.
- 파이썬 표준 라이브러리 문서. <https://docs.python.org/>에서 2021. 07. 13. 검색.
- 파이썬 표준 라이브러리 상세 항목. <https://docs.python.org/ko/3/library/index.html>에서 2021. 07. 13. 검색.
- GitHub에서 많이 사용되고 있는 프로그래밍 언어. <https://octoverse.github.com/>에서 2021. 06. 10. 검색.
- OWASP Top Ten. <https://owasp.org/www-project-top-ten/>에서 2021. 08. 16. 검색.
- PMD 다운로드 및 사용법. <https://pmd.github.io/>에서 2021. 07. 26. 검색.

NCS학습모듈 개발이력

발행일	2021년 12월 31일		
세분류명	응용SW엔지니어링(20010202)		
개발기관	(사)한국정보통신기술사협회(개발책임자: 온기현), 한국직업능력연구원		
집필진	김승환(㈜캐롯아이)*	안응원(미라콤아이앤씨)	
	김원기(LG CNS)	유현주(한국전산감리원)	
	박주형(㈜오아시스비즈니스)	검토진	홍필두(한국폴리텍대학)
	엄기영(우리FIS)		
	윤혜경(만도)		
	최홍선(㈜에이스기술단)	*표시는 대표집필자임	

프로그래밍 언어 응용(LM2001020230_19v4)

저작권자	교육부
연구기관	한국직업능력연구원
발행일	2021. 12. 31.
ISBN	979-11-339-9443-4

※ 이 학습모듈은 자격기본법 시행령(제8조 국가직무능력표준의 활용)에 의거하여 개발하였으며, NCS통합포털사이트(<http://www.ncs.go.kr>)에서 다운로드 할 수 있습니다.



www.ncs.go.kr