프로그래밍 언어 활용

1. **구조적 프로그래밍 언어 활용하기**

1 - 1 구조적 프로그래밍 설계

1. 구조적 프로그래밍 설계 구조 및 절차

**1. 구조적 프로그래밍 설계 구조**

**(1) 순차구조**

프로그램 코드가 위에서 아래로 일직선으로만 실행되는 구조로 실행을 위하여 작은 단위로

나눈 후 순서대로 진행되는 구조이다.

**(2) 선택구조**

선택구조는 분기형 구조로서, 주어진 조건 만족 여부에 따라 처리 대상과 순서를 전달하는 구조이다.

어떤 것을 선택하느냐에 따라 실행 순서가 달라지며 조건문 (if, if-else)를 이용하여 실행한다.

특정 조건이 들어와 참이냐 거짓이냐에 따라 처리가 분기되는 구조이다.

**(3) 반복구조**

반복형 구조로 조건이 만족할 때까지 동작을 반복하는 구조이다. 반복문(while, for)을 사용하여

실행한다. 참인 조건이 될 때까지 진행되는 구조이다.

1. - 2 구조적 프로그래밍 언어 활용
   1. 구조적 프로그래밍 언어 개요
   2. **절차식 언어**

구조적 프로그래밍은 프로그램이 실행될 때 위에서 아래로 순서대로 실행되는 방식의 프로그램이다.

구조적 프로그래밍의 이전의 프로그래밍에서는 Goto문을 사용함으로써 진행 방향이 일괄적이지

않았던 단점을 개선한 언어이다.

* 1. **명령어 언어**

프로그램은 순차적으로 실행되면서 필요한 함수를 호출하거나, 실행을 시킬 수 있는데 명령어들을

이용하여 진행되어 진다. 명령은 메인에서 서브프로그램으로 전달되는데 하향식 방식을 이용한다.

* 1. **함수중심 언어**

구조화를 시켜가면서 독립적인 행위를 할 수 있는 작은 단위로 나뉘는데, 이때 함 수를 이용하여

구성하게 된다. 이런 함수는 서브프로그램이 될 수 있다

* 1. 구조적 프로그래밍 유형
     1. **c언어 2. 파스칼(Pascal) 3.에이다(Ada)**

1. **객체지향 프로그래밍 언어 활용하기**

2 - 1 객체지향 프로그래밍 설계

1. 객체지향 설계 원칙의 이해

**1. 객체지향 속성**

**(1) 캡슐화**

캡슐화는 객체의 속성과 행위를 하나로 묶고, 실제 구현 내용 일부를 외부에 감추어 은닉하는

기법이다. 외부 객체는 객체 내부의 구조를 알지 못하게 하고, 내부 객체에서 제어하여 제공하는

필드와 메소드만 이용할 수 있다.

**(2) 추상화**

추상화는 어떤 실체로부터 공통적인 부분이나 관심 있는 특성들만 하나로 모은 것을 의미한다.

어떤 하위 클래스들에 존재하는 공통적인 메소드를 인터페이스로 정의하는 것도 이에 포함된다.

**(3)** **다형성**

다형성은 같은 모양의 함수가 상황에 따라 다르게 동작하는 것을 의미한다. 다형성을 이용하는

기법은 오버로딩과 오버라이딩이 있다.

(가) 오버로딩 : 함수명은 같으나 함수의 매개변수 숫자, 타입 등을 다르게 해서 사용하는 것

(나) 오버라이딩 : 상위 클래스의 메소드를 하위 클래스에서 똑같은 이름으로 재정의하는 것

**(4) 정보 은닉**

캡슐화 정보들을 밖에서 접근 불가능하도록 하는 것을 은닉화라고 한다.

property(변수)와 메소드(함수)가 하나의 캡슐 안에 묶인다.

**(5) 상속성**

부모 클래스에서 새로운 기능을 추가해 자식 클래스를 만들어 사용하는 기법을 상속성이라고 한다.

자식 클래스에서 따로 정의하지 않아도 부모 클래스에서 정의된 것을 자동으로 상속받아 구현가능

**2. 객체지향 설계 원칙**

**(1) 단일 책임 원칙**

모든 클래스는 각각 하나의 책임만 가져야 한다. 클래스는 그 책임을 완전히 캡슐화해야 함.

**(2) 개방 폐쇄 원칙**

확장에는 열려 있고 수정에는 닫혀 있어 기존의 코드를 변경하지 않고(Closed),

기능을 추가할 수 있도록(Open) 설계가 되어야 한다는 원칙을 말한다.

**(3) 리스코프 치환 원칙**

자식 클래스는 자신의 부모 클래스를 대체할 수 있다는 원칙으로 부모 클래스가 들어갈 자리에

자식 클래스를 넣어도 잘 작동해야 한다. 자식은 부모 클래스의 확장만 수행하도록 해야 만족한다.

**(4) 인터페이스 분리 원칙**

한 클래스는 자신이 사용하지 않는 인터페이스는 구현하지 말아야 한다는 원칙이다.

하나의 평범한 인터페이스보다 여러 개의 구체적인 인터페이스가 좋다

**(5) 의존 역전 원칙**

의존 역전 원칙은 변화가 없는 것에 의존해야 한다는 원칙이다. 구체적인 클래스 보다 인터페이스나

추상 클래스와 관계를 맺어야 하는 원칙이다

2 - 2 객체지향 프로그래밍 언어 활용

1. 객체지향 프로그래밍 언어의 개요
2. **객체지향 프로그래밍 언어 구성요소**

**(1) 클래스(Class) (2) 객체(Object) (3) 메소드(Method) (4) 속성(Property)**

1. 디자인 패턴
2. **디자인 패턴 구조**

**(1) 콘텍스트(Context)**

문제가 발생하는 여러 상황을 기술한다. 패턴이 적용될 수 있는 상황을 나타내고,

패턴이 유용하지 못한 상황을 나타내기도 한다.

**(2) 문제(Problem)**

패턴이 적용되어 해결될 필요가 있는 여러 디자인 이슈들을 기술한다.

이때에는 여러 제약 사항과 영향력도 문제 해결을 위해 고려해야 한다.

**(3) 해결(Solution)**

문제를 해결하도록 설계를 구성하는 요소들과 그 요소들 사이의 책임 관계, 협력 관계를 기술한다.

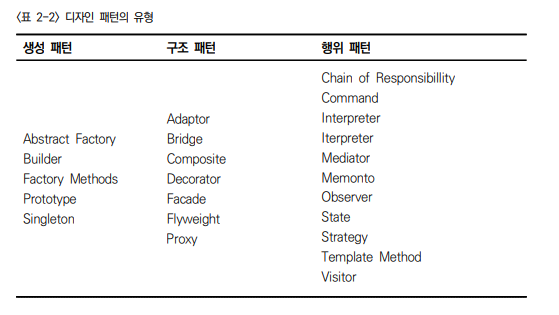
해결은 다양한 상황에 적용할 수 있는 일종의 템플릿이다

1. **디자인 패턴의 종류**

**(1) GoF 디자인 패턴**

GoF 디자인 패턴은 23가지의 디자인 패턴이 정리되어 있고, 구현 방식에 따라 생성(Creational),

구조(Structural), 행위(Behavioral) 3가지로 분류한다



1. **스크립트 언어 활용하기**

3 - 1 스크립트 언어 설계

1. 스크립트 언어 개요

스크립트 언어(Scripting Language)는 응용소프트웨어를 제어하는 컴퓨터 프로그래밍 언어이다. 초창기

스크립트 언어는 배치 언어(Batch Languages) 또는 작업 제어 언어(Job Control Language)라고도 불렸다

1. **스크립트 언어 특징**

스크립트 언어는 응용프로그램과 독립하여 사용되고 다른 응용프로그램의 언어와 다른 언어로

사용되어, 최종 사용자가 응용프로그램의 동작을 사용자의 요구에 맞게 수행할 수 있도록 해준다

**(1) 인터프리터**

언어 코드를 작성함과 동시에 인터프리터가 기계어로 번역하고 실행한다.

**(2) 단순한 구문**

스크립트 언어는 타 프로그래밍 언어에 비해 단순한 구문과 의미를 내포한다.

**(3) 컴파일 시간 소요**

스크립트 언어는 컴파일된 프로그램보다 실행 시간이 오래 걸린다. 스크립트 언어는 모든 명령어가

기본 명령어 처리기에 의해 직접 처리되지 못하고, 다른 프로그램에 의해 전 처리가 필요하기 때문.

**(4) 신속한 활용**

일반적으로 스크립트 언어는 빠르게 배우고 작성하기 위해 고안된 언어이다.

3 - 2 스크립트 언어 활용

1. 스크립트 언어의 유형과 원리
2. **스크립트 프로그래밍 언어 유형**

**(1) JavaScript (자바스크립트) (2) jQuery (3) JSP (JavaServer Pages)** ex) <% %>

**(4) PHP (Hypertext Preprocessor)**

HTML 문서 안에 포함하여 작동하는 서버 측 스크립트 프로그래밍 언어이다. 리눅스 운영체제에

아파치 웹 서버를 설치하고 MySQL DB 환경에서 주로 같이 사용된다.

**(5) ASP (Active Server Pages)**

마이크로소프트사에서 개발된 윈도 서버에서 운영할 수 있도록 개발된 스크립트 방식의

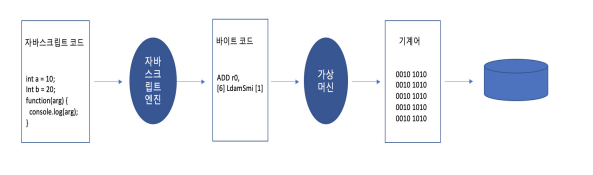
웹 프로그래밍 언어이다.

**(6) Python**

**(7) VBScript**

미국 마이크로소프트 Visual Basic(비주얼베이직) 기반의 스크립트 프로그래밍 언어이다.

**2. 스크립트 프로그래밍 원리**



**(1) 스크립트 코드 작성**

최초 스크립트 코드를 작성한다.

**(2) Byte Code 변환**

스크립트 엔진은 위에서부터 한 줄씩 해석하면서 가상 머신이 이해할 수 있는 Byte Code로 변환한다.

**(3) 기계어 변환**

가상 머신은 Byte Code를 CPU가 이해할 수 있는 기계어로 변환한다. 단, CPU마다 기계어를 다르게

해석하므로 가상 머신은 CPU별로 최적화된 기계어를 만들어낸다.

**(4) CPU 코드 변환**

컴퓨터의 CPU는 가상 머신이 만들어낸 기계어를 수행한다. 기계어대로 메모리에 데이터를 저장하고,

읽고, 연산하고, 출력하는 작업 등을 수행한다.