



알기 쉽게 해설한
자바 프로그래밍 10판

Chapter 01. 컴퓨터와 프로그램 그리고 자바

학습목표

- 우리가 컴퓨터를 통해서 할 수 있는 것이 무엇인지 알아봅니다.
- 컴퓨터에서 실행되는 프로그램은 어떻게 만들어지는가를 알아봅니다.
- 우리가 만든 프로그램이 컴퓨터에서 어떻게 실행되는지 알아봅니다.
- 자바 언어는 어떻게 탄생하였고, 어떤 환경을 가지고 있는지 알아봅니다.
- 현재 많은 프로그래밍 언어들의 사용 분포에 대해서 학습합니다.
- 자바 언어의 특징에 관해 학습합니다.
- 자바 가상기계(JVM)에 관해 학습합니다.

목차

Section 1. 컴퓨터는 무엇을 실행하는가?

Section 2. 프로그램은 어떻게 만들어지는가?

Section 3. 프로그램은 컴퓨터에서 어떻게 실행되는가?

Section 4. 프로그래밍 언어 Java

Section 1.

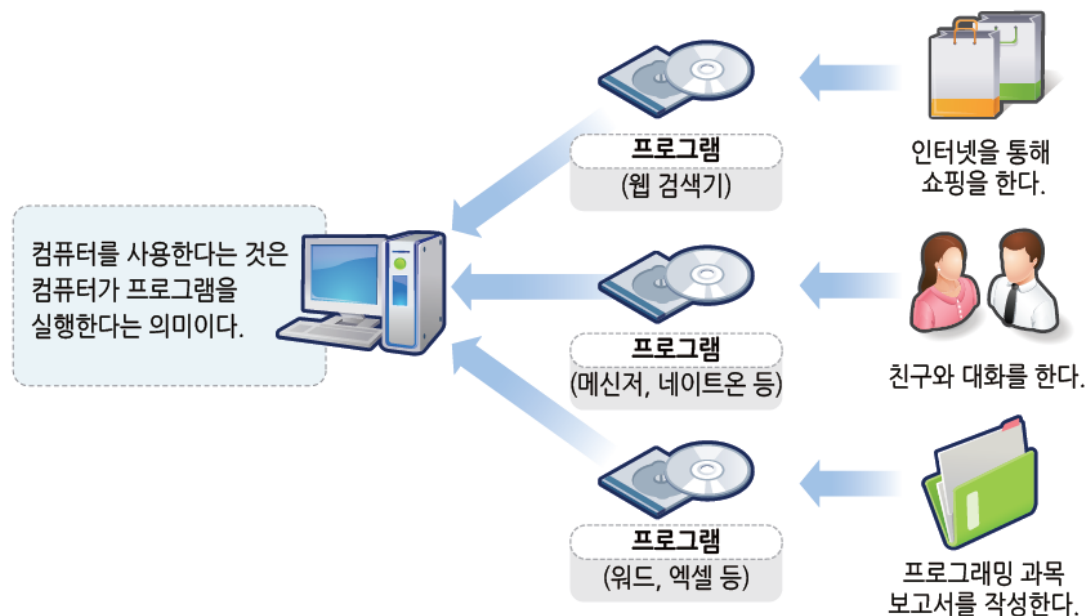
컴퓨터는 무엇을 실행하는가?

1 컴퓨터는 무엇을 실행하는가?

1-1 컴퓨터가 실행하는 것

- 컴퓨터를 통해 쇼핑을 한다.
- 컴퓨터를 통해 친구와 대화를 한다.
- 컴퓨터를 통해 보고서를 작성한다.

➤ 우리는 컴퓨터를 통해 다양한 일들을 하고 있지만, 공통적인 것은 컴퓨터가 소프트웨어(프로그램)를 실행한다는 것



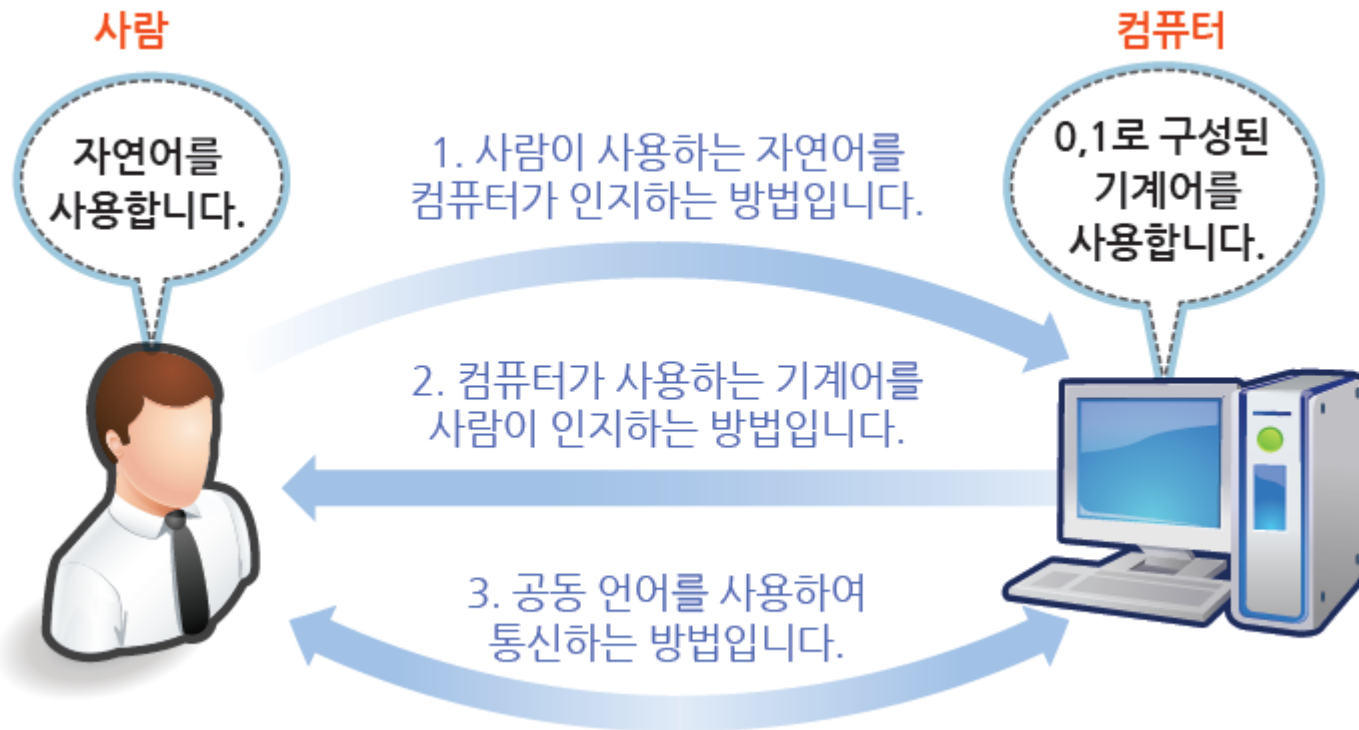
Section 2.

프로그램은 어떻게 만들어지는가?

2 프로그램은 어떻게 만들어지는가?

2-1 컴퓨터와 사람의 상호작용

● 사람과 컴퓨터의 상호작용





2 프로그램은 어떻게 만들어지는가?

2-1 컴퓨터와 사람의 상호작용

- **사람의 언어를 컴퓨터가 인지**

- 많은 과학자들에 의해 사람들이 사용하는 자연어(Natural Language)를 컴퓨터가 인식하기 위한 노력이 인공지능(Artificial Intelligence) 분야에서 이루어졌음

- **컴퓨터 언어를 사람이 사용**

- 컴퓨터 언어인 2진법만을 사용해서 컴퓨터를 사용한다는 것은 사람에게는 너무나도 힘들고 어려운 일임

- **사람과 컴퓨터의 공동언어 사용(프로그래밍 언어)**

- 컴퓨터와 사람이 상호작용을 하기 위해서는 반드시 프로그래밍 언어를 사용해야 함

2 프로그램은 어떻게 만들어지는가?

2-2 프로그래밍 언어

● 프로그래밍 언어의 발전 단계에 따른 세대별 구분





2 프로그램은 어떻게 만들어지는가?

2-2 프로그래밍 언어

- 주요 언어로 작성된 프로그램(1부터 100까지의 합을 구하는 프로그램)

- 어셈블리 프로그램

01: TEST	START	0	
02: BEGIN	BALR	BASE,0	
03:	USING	BEGIN+2,BASE	
04:	SR	4,4	← 4번 레지스터의 내용을 0으로 설정
05:	L	2,=F'1'	← 2번 레지스터에 1을 넣는다
06:	L	3,=F'100'	← 3번 레지스터에 100을 넣는다
07: LOOP	AR	4,3	← 4번과 3번의 내용을 더한다
08:	SR	3,2	← 3번에서 2번의 내용을 뺀다
09:	BCT	3,*-4	← 07번 문장으로 분기한다
10:	BR	14	
11:	LTORG		
12: BASE	EQU	15	
13:	END		



2 프로그램은 어떻게 만들어지는가?

2-2 프로그래밍 언어

● 주요 언어로 작성된 프로그램(1부터 100까지의 합을 구하는 프로그램)

- 포트란 프로그램

```
01:      ISUM=0
02:      DO 100 I=1,100,1 ←----- 1부터 100까지 1씩 증가시켜 가면서 100(04번 문장)번까지를 반복
03:          ISUM=ISUM+I ←----- 합계에 더한다
04: 100    CONTINUE
05:      WRITE (6,200) ISUM ←-----
06: 200    FORMAT(8X,I5) ←----- 합계를 출력한다
07:      STOP
08:      END
```

● 주요 언어로 작성된 프로그램(1부터 100까지의 합을 구하는 프로그램)

- 코볼 프로그램

```

13: WORKING-STORAGE SECTION.
14: 01 LIST.
15:      03 N PIC 9(3).
16:      03 HAP PIC 9(4).
17: PROCEDURE DIVISION.
18: AA.      PERFORM BB UNTIL N = 100.
19:          DISPLAY "***1부터 100까지의 합***"
20:          DISPLAY "HAP = " HAP.
21:          STOP RUN.
22: BB.      ADD 1 TO N.
23:          ADD N TO HAP.
  
```

N이 100이 될 때까지 BB 블록을 반복 수행
 합계를 출력
 프로그램 종료
 N값을 1 증가
 N값을 HAP에 더한다



2 프로그램은 어떻게 만들어지는가?

2-2 프로그래밍 언어

- 주요 언어로 작성된 프로그램(1부터 100까지의 합을 구하는 프로그램)

- 베이직 프로그램

```
01: 10 SUM = 0
02: 20 FOR I=1 TO 100
03: 30     SUM = SUM + I
04: 40 NEXT I
05: 50 PRINT SUM
06: 60 END
```

← I가 1부터 100이 될 때까지 04번 문장까지 반복

← SUM에 I 값을 더한다

← 반복의 범위

← SUM 출력



2 프로그램은 어떻게 만들어지는가?

2-2 프로그래밍 언어

- 주요 언어로 작성된 프로그램(1부터 100까지의 합을 구하는 프로그램)

- 파스칼 프로그램

```
01: program hap;
02: uses  Crt;
03: var ←----- 변수를 선언한다
04:   i,sum : integer; ←-----
05: begin
06:   ClrScr;
07:   sum := 0; ←----- 변수를 0으로 설정
08:   for i := 1 to 100 do ←----- 1부터 100까지 반복하며
09:     sum := sum + i; ←----- sum에 값을 더한다
10:   writeln('1부터 100까지의 합은 ==> ',sum); ←----- 값을 출력한다
11: end.
```



2 프로그램은 어떻게 만들어지는가?

2-2 프로그래밍 언어

- 주요 언어로 작성된 프로그램(1부터 100까지의 합을 구하는 프로그램)

- C 프로그램

```
01: #include <stdio.h>
02:
03: main()
04: {
05:     int i, sum; ←----- 변수를 선언한다
06:     clrscr();
07:     sum=0; ←----- sum 변수를 0으로 설정
08:     for ( i = 1; i <= 100; i++) ←----- 1부터 100까지 반복하며
09:         sum = sum + i; ←----- sum에 값을 더한다
```



2 프로그램은 어떻게 만들어지는가?

2-2 프로그래밍 언어

● 주요 언어로 작성된 프로그램(1부터 100까지의 합을 구하는 프로그램)

- Java 프로그램

```
01: public class SumTest {  
02:     public static void main(String[] args) {  
03:         int i, sum=0;   
04:         for ( i = 1 ; i <= 100 ; i = i + 1 ) {  
05:             sum = sum + i;  
06:         }  
07:         System.out.println("1부터 100까지의 합은 " +sum);  
08:     }  
09: }
```

변수를 선언하고,
sum을 0으로 설정한다

1부터 100까지
반복하며 sum에
값을 더한다

값을 출력한다



2 프로그램은 어떻게 만들어지는가?

2-2 프로그래밍 언어

- 주요 언어로 작성된 프로그램(1부터 100까지의 합을 구하는 프로그램)

- 파이썬 프로그램 (13p.)

```
01: sum=0 ← sum을 0으로 설정한다
02: for i in range(1,101) : ←
03:     sum = sum + i ← 1부터 100까지 반복하며 sum에 값을 더한다
04: print(sum) ← 값을 출력한다
```

- 앞에서 살펴본 8가지 종류의 언어에서 느낄 수 있는 점은 몇 종류의 언어(어셈블리, 코볼)를 제외하고는 대부분 구조가 유사

Section 3.

프로그램은 컴퓨터에서 어떻게 실행되는가?

3 프로그램은 컴퓨터에서 어떻게 실행되는가?

3-1 컴파일(compile) 기법

- 프로그램이 컴파일러(compiler)에 의해 0과 1로 구성된 이진 파일(0과 1로 구성된 파일)로 번역된 다음, 번역된 파일이 컴퓨터에서 실행되는 기법



3 프로그램은 컴퓨터에서 어떻게 실행되는가?

3-2 인터프리트(interprete) 기법

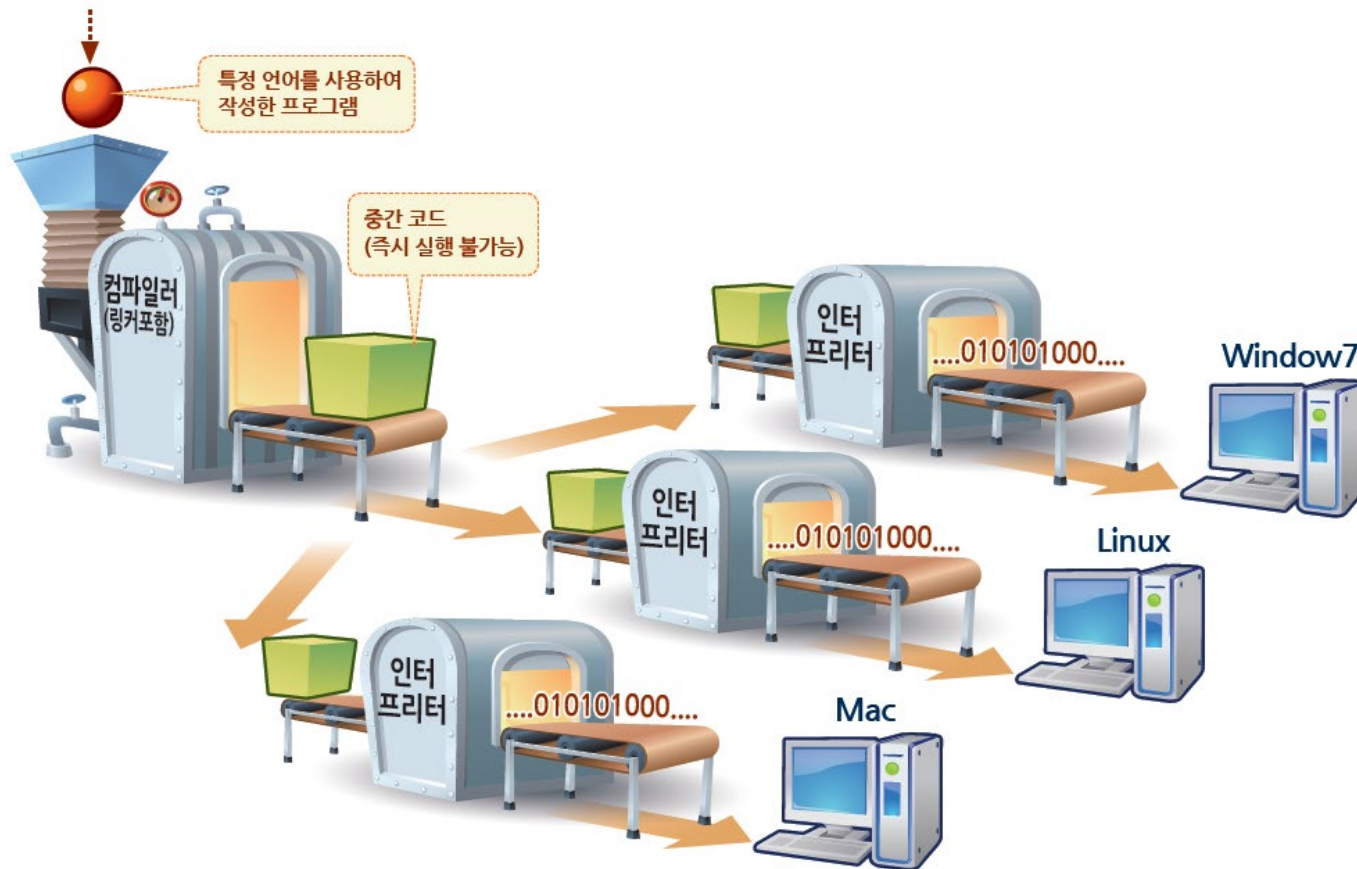
- 인터프리트(해석) 기법은 인터프리터(interpreter)에 프로그램을 실행시키는 방법



3 프로그램은 컴퓨터에서 어떻게 실행되는가?

3-3 하이브리드(hybrid) 기법

- 컴파일 기법과 인터프리터 기법을 모두 사용하는 방식



Section 4.

프로그래밍 언어 Java



4 프로그래밍 언어 Java

4-1 자바의 개요

- 자바 언어 : 오크(Oak)라는 언어로부터 탄생
- 오크 언어 : 1991년 미국 선(Sun) 마이크로시스템사의 컴퓨터 과학자인 제임스 고슬링(James Gosling)에 의해 개발된 언어
 - 가전제품의 기능을 프로그램으로 제공하기 위해 개발
 - 제임스 고슬링은 앞으로 많은 가전제품(전화, TV, 비디오, 컴퓨터 등)들이 하나의 거대한 네트워크에 연동될 것으로 예측하였고, 또한 제품들의 기능을 제공하는 내장된 프로그램(embedded program)들이 부품(component)처럼 사용될 것이라고 예측
 - 많은 가전사들의 무관심과 현실성에 대한 문제 때문에 별로 관심을 끌지 못함



4 프로그래밍 언어 Java

4-1 자바란?

● 자바 언어

- 오크 언어의 많은 개념들을 그대로 이어받았음
- C/C++에서 어렵게 사용되는 포인터나 메모리 조작 등의 개념을 과감하게 제거하거나 개선
- 신뢰성을 증대시키기 위해 예외 처리(exception-handling) 기능을 효율적으로 제공하여 예상치 못한 오류 등을 처리하는 방법을 제공
- 1995년 JAVA 베타 2 버전이 탄생
- 1996년 JAVA 1.0이 공식 발표
- 현재 JAVA 1.2 사용



4 프로그래밍 언어 Java

4-1 자바란?

- 자바 개발자 제임스 고슬링(James Gosling)과 자바 엠블렘





4 프로그래밍 언어 Java

4-1 자바란?

● 자바의 사용 : 3가지 종류의 플랫폼 제공

- Java 2 ME(Mobile Edition) : PDA나 스마트폰 등 소형 기기를 위한 개발 환경
- Java 2 SE(Standard Edition) : 클라이언트 중심의 일반적인 자바 응용 프로그램 개발 환경
- Java 2 EE(Enterprise Edition) : 서버 중심의 기업용 소프트웨어 개발 환경



4 프로그래밍 언어 Java

4-1 자바란?

● 2020년대 들어 현재까지 가장 많이 사용되는 프로그래밍 언어

Aug 2020	Aug 2019	Change	Programming Language	Ratings	Change
1	2	⬆	C	16.98%	+1.83%
2	1	⬇	Java	14.43%	-1.60%
3	3		Python	9.69%	-0.33%
4	4		C++	6.84%	+0.78%
5	5		C#	4.68%	+0.83%
6	6		Visual Basic	4.66%	+0.97%
7	7		JavaScript	2.87%	+0.62%
8	20	⬆	R	2.79%	+1.97%
9	8	⬇	PHP	2.24%	+0.17%
10	10		SQL	1.46%	-0.17%
11	17	⬆	Go	1.43%	+0.45%
12	18	⬆	Swift	1.42%	+0.53%
13	19	⬆	Perl	1.11%	+0.25%
14	15	⬆	Assembly language	1.04%	-0.07%
15	11	⬇	Ruby	1.03%	-0.28%



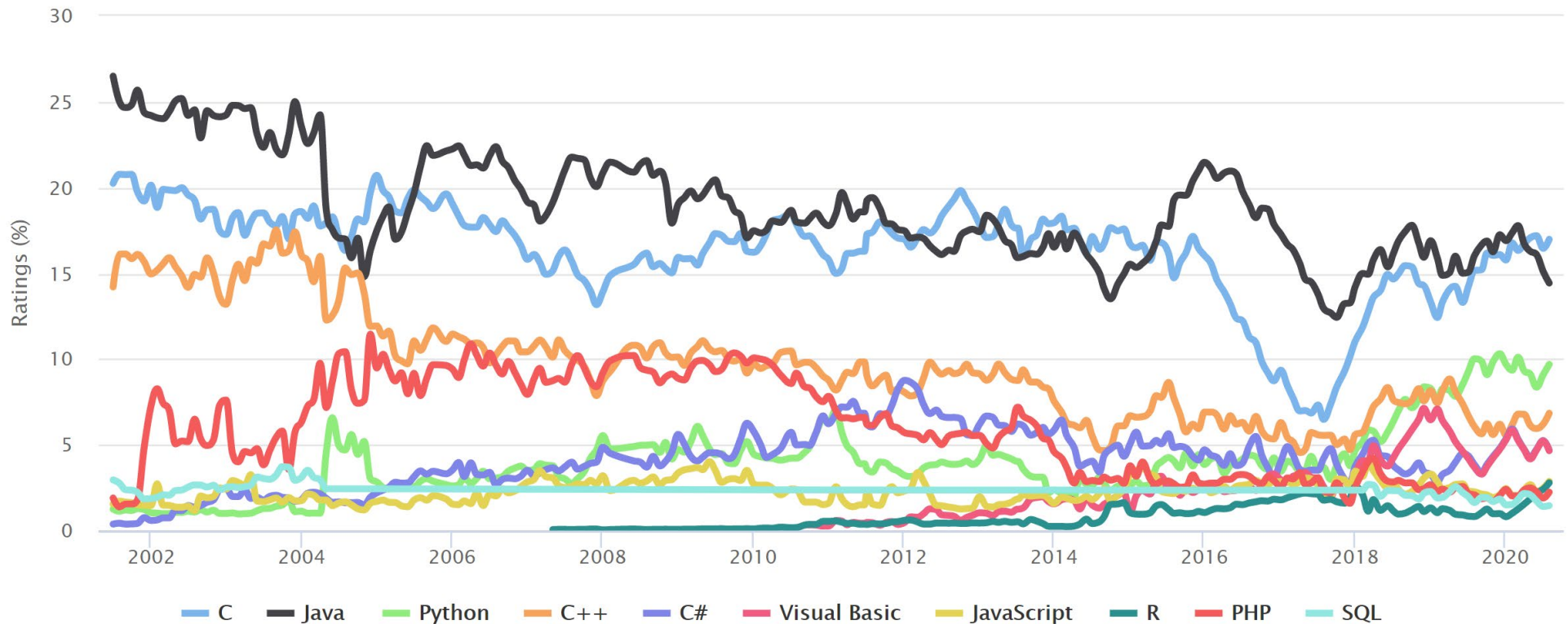
4 프로그래밍 언어 Java

4-1 자바란?

● 2000년대 프로그래밍 언어 사용 순위의 변화

TIOBE Programming Community Index

Source: www.tiobe.com





4 프로그래밍 언어 Java

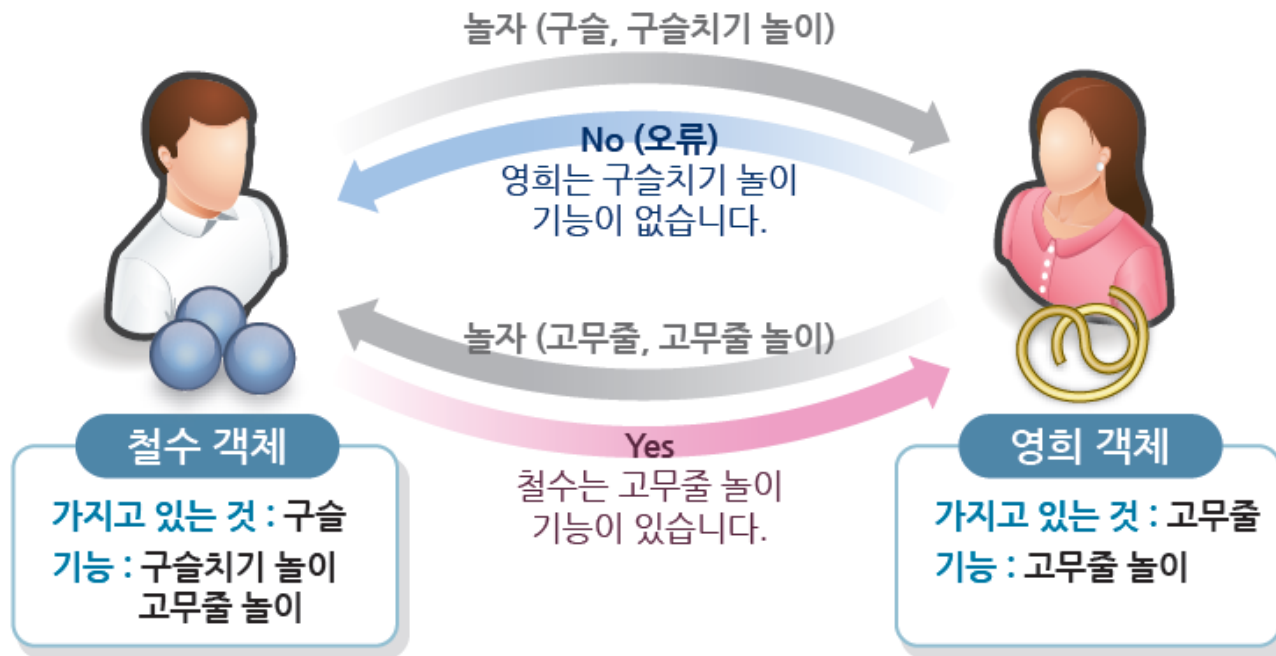
4-2 자바 언어의 특징

1) 객체지향언어(Object Oriented Language)

- 자바는 완전한 객체지향 언어
- 객체지향의 특성인 클래스, 상속, 캡슐화, 다형성 등의 개념이 잘 적용된 언어
- 객체지향 프로그래밍은 우리가 살아가는 실 세계와 동일한 사고방식의 프로그램
- 자바 언어는 가장 쉽게 이해할 수 있으며, 빨리 배울 수 있는 프로그래밍 언어

4-2 자바 언어의 특징

1) 객체지향 언어 : 실 세계 객체 지향의 예

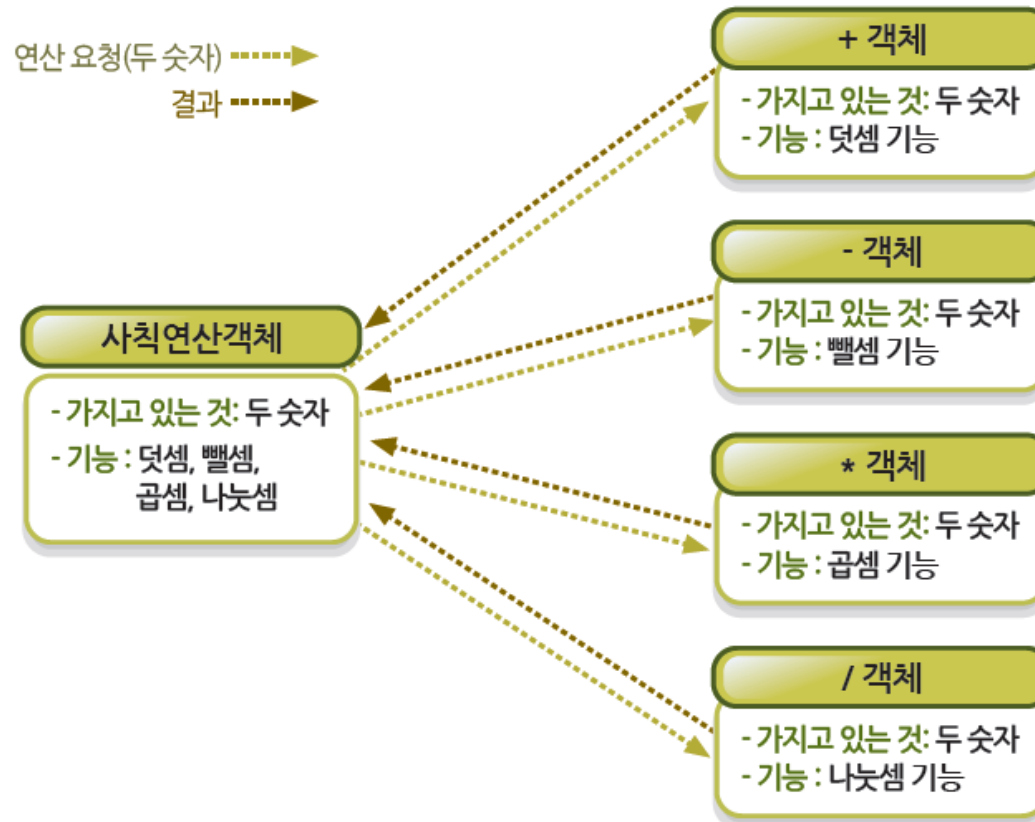




4 프로그래밍 언어 Java

4-2 자바 언어의 특징

1) 객체지향 언어 : 자바의 객체 지향의 예



4-2 자바 언어의 특징

2) 운영 체제에 독립적 : 소프트웨어와 운영체제

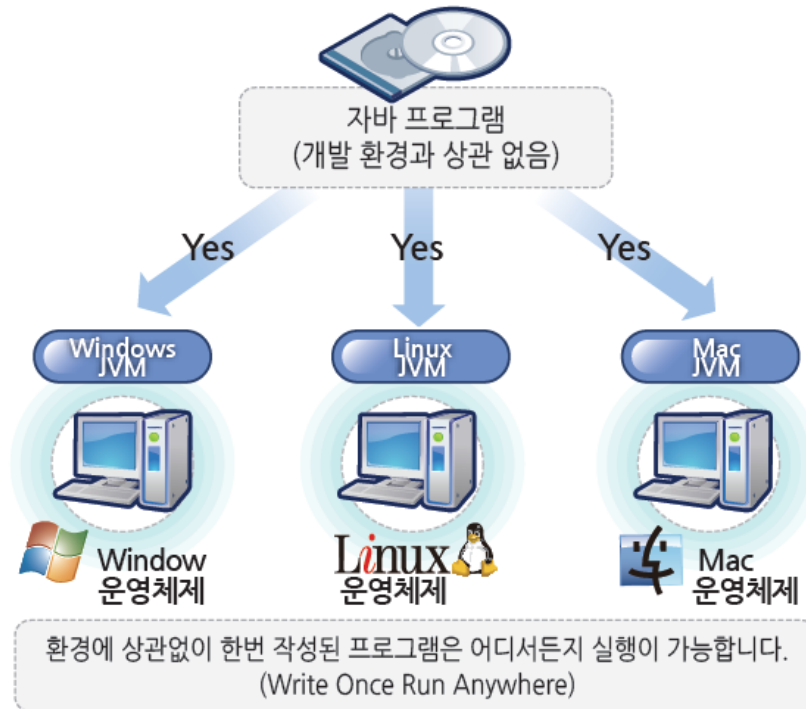
- 컴퓨터의 운영체제 : 하드웨어를 관리하는 시스템 소프트웨어
- 일반적으로 컴퓨터에서 실행되는 소프트웨어 : 운영체제에 종속적
- 윈도 시스템에서 C 언어로 개발된 소프트웨어는 윈도 시스템에서만 실행



4-2 자바 언어의 특징

2) 운영체제에 독립적

- 자바 프로그램 : JVM(Java Virtual Machine : 자바가상머신)이 구축된 컴퓨터에서는 어디에서든지 실행시킬 수 있음





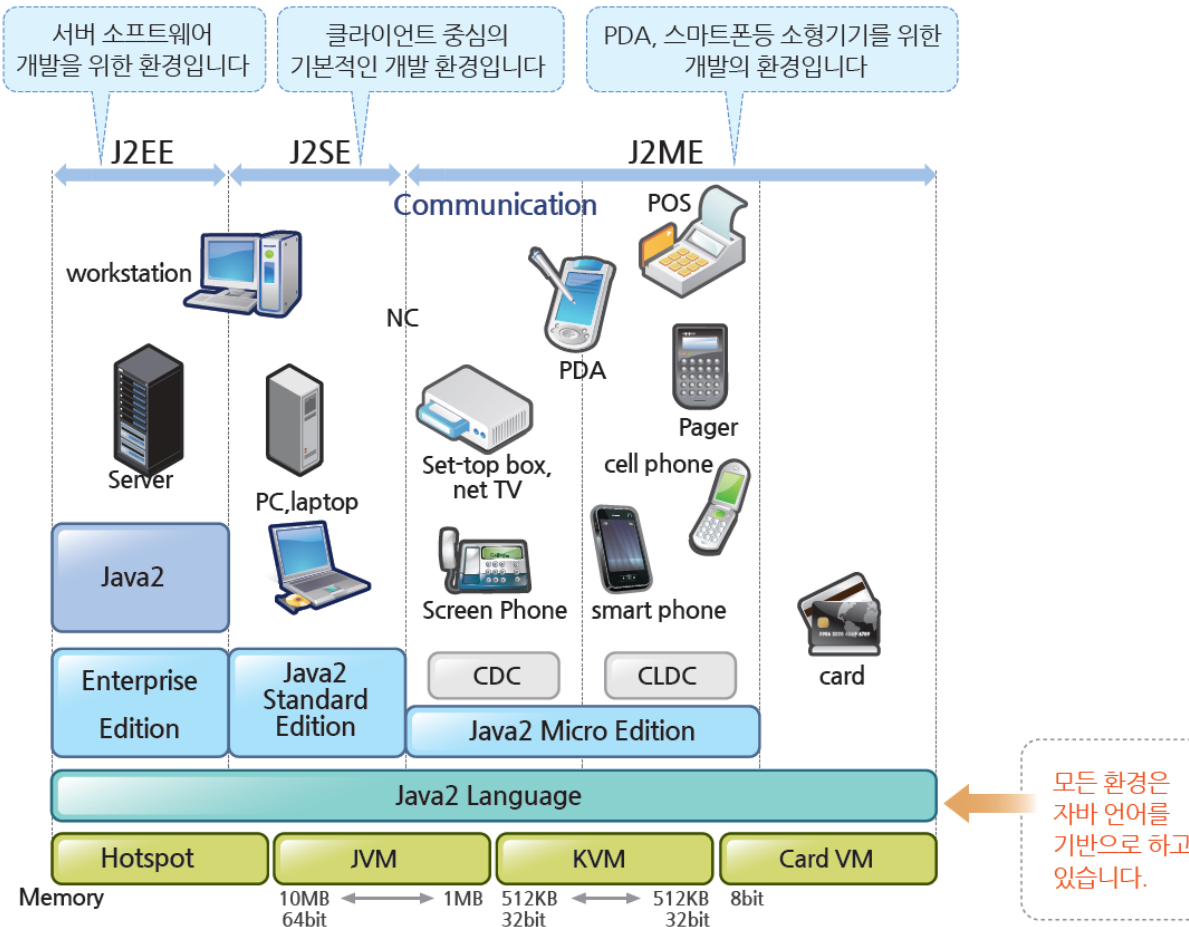
4 프로그래밍 언어 Java

4-2 자바 언어의 특징

3) 무료 개발 환경

- 자바의 모든 개발 환경 : 개방된 형태를 취하고 있음
- 자바를 개발한 선사는 현재까지도 자바의 모든 개발 환경을 무료로 제공
- 지속적으로 최신 버전을 제공
- 현재 선사는 3종류의 개발 환경을 제공

3) 무료 개발 환경 : 선사에서 제공하는 플랫폼의 종류





4 프로그래밍 언어 Java

4-2 자바 언어의 특징

4) 무수한 라이브러리 : 대규모의 라이브러리 클래스 제공

The screenshot displays the Java Platform, Standard Edition 8 API Specification website. The left sidebar lists various packages and classes. The main content area shows the 'Overview' section, which includes a list of packages and their descriptions.

Package	Description
java.applet	Provides the classes necessary to create an applet and the classes an applet uses to communicate with its applet context.
java.awt	Contains all of the classes for creating user interfaces and for painting graphics and images.
java.awt.color	Provides classes for color spaces.
java.awt.datatransfer	Provides interfaces and classes for transferring data between and within applications.
java.awt.dnd	Drag and Drop is a direct manipulation gesture found in many Graphical User Interface systems that provides a mechanism to transfer information between two entities logically associated with presentation elements in the GUI.



4 프로그래밍 언어 Java

4-3 자바의 버전

- 자바는 1996년 JDKJava Development Kit 1.0을 시작
- 현재는 Java 11,12에 이르기까지 지속적으로 새로운 버전이 출시

버전명	연도	특징
JDK 1.0 ~ JDK 1.4	1996 ~ 2002	<ul style="list-style-type: none"> • Java Applet, RMI, JDBC, reflection, JIT, Inner Class 개념 도입 • 1.2 : 자바를 세가지 버전으로 발표(Java SE 1.2, ME 1.2, EE 1.2) • Swing, Corba IDL, Collection Framework 포함 • HotSpot JVM 포함 • JNDI(Java Naming and Directory Interface) 포함 • 1.4 : JCP에 의해서 오픈소스 정책으로 자바가 관리되기 시작한 버전 • 1.4 : Java 2 Security 모델의 확립(Sandbox) • Language: assert 도입
Java 5	2004	<ul style="list-style-type: none"> • 기능적으로 가장 많은 변화, Java 5로 불림. • Language : Generics , annotation, auto boxing, enum,vararg ,foreach, static imports 도입 • API : java.util.concurrent API, Scanner class
Java 6	2006	<ul style="list-style-type: none"> • JVM/Swing에 있어 많은 Performance 향상(synchronization, compiler, GC, start-up time) • GI(Garbage First) GC도입.
Java 7	2011	<ul style="list-style-type: none"> • JVM : Dynamic Language support(invokedynamic – new byte operation) • Language : Switch에서 String, try-resource, generics에서 타입추론, 숫자에서 underscore사용 • API : Concurrency, NO, sort, crypto, GPU 강화 • JavaFX가 기본으로 포함
Java 8	2014	<ul style="list-style-type: none"> • 오라클로 인수된 후 첫 번째 버전 • Java5 이후 가장 큰 언어적 변화(Lambda 및 함수형프로그래밍, default method 도입) • Language : Lambda expression, Default Method Interface(다중상속지원) 등 • API : Nashorn(JS엔진), new Date and Time API, stream api, Collection에 대한 함수형화 • 병렬처리에 적합한 구조로 진화
Java 9	2017	<ul style="list-style-type: none"> • Modular System (Jigsaw)지원예정 • Money API지원예정 • Java Shell지원예정 • 변수에 대한 타입 추론 지원예정(var, val) • OpenCL이용한 자동화된 병렬 프로그래밍 지원예정 • value 타입 지원예정

버전명	연도	특징
Java 10	2018	<ul style="list-style-type: none"> • var 지역변수 타입 추론 • 병렬처리 가베지 컬렉터 • 개별 Thread로 분리된 Stop-The World • 루트 CA
Javatl, 12	2019	<ul style="list-style-type: none"> • switch 문 확장 • lambda 파라미터에 대한 지역변수 문법 • 애플론 가베지 컬렉터 • HTTP 클라이언트 표준화 • Oracle JDK 유료 모델로 전환(비영리 기관 사용 가능) • 무료로 사용하는 OpenJDK 등장

● 컴퓨터 개발 초기 : 하드웨어 구조를 숙지하고 프로그래밍

```

LD2  0,2(RLINK)
J2N  1F
ST5   0,3(RLINK)
JMP   2B
1H    ENN2 0,2
D5    ENT1 -Y,2
LD4   0,2(LLINK)
LD6   0,4(RLINK)
J1NZ  D2
ST5   DY(LLINK)
ENNA  DY
  
```



A회사의 하드웨어

```

D1    STJ  9F
LD4   Y(LLINK)
1H    ENT2 0,4
2H    LD4  0,2(LLINK)
J4NZ  1B
D2    LD1  0,2(TYPE)
JMP   **+1,1
JMP   CONSTANT
JMP   VARIABLE
JMP   LN
JMP   NEG
  
```



B회사의 하드웨어

● 운영체제의 등장 : 운영체제에 기반한 프로그래밍

```
while(L < R){
    while((a[L] <= a[pivot]) && (L <= R)) L++;
    while((a[R] > a[pivot]) && (L <= R)) R--;
    if(L <= R){
        temp = a[L];
        a[L] = a[R];
        a[R] = temp;
    }
}
```



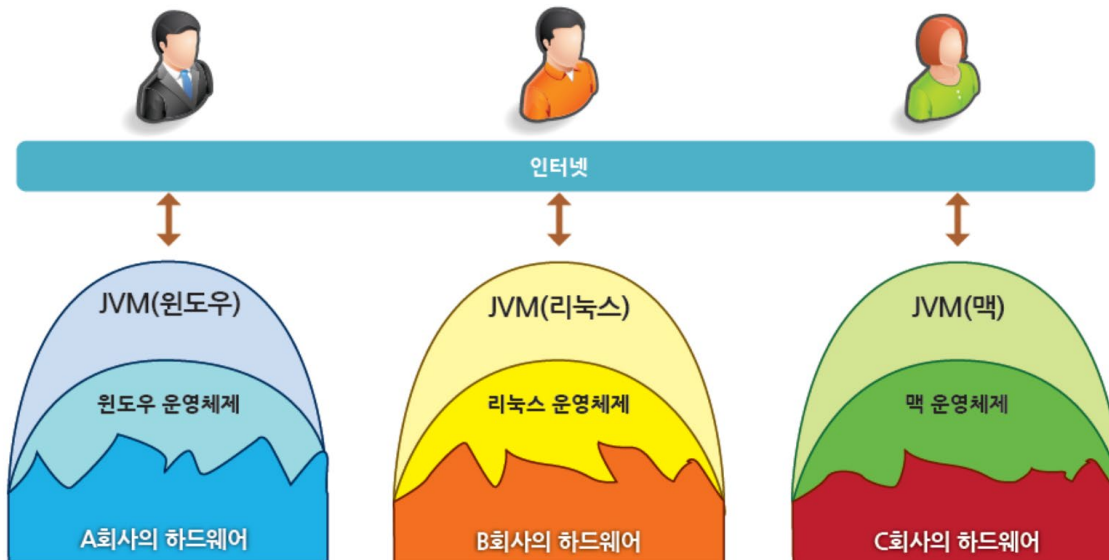
* 운영체제가 다를 경우 프로그램 다시 작성(컴파일)

- 자바 가상 기계 : “자바 프로그램이 실행되는 컴퓨터”
- 서로 다른 운영체제 상에서도 자바 가상기계에서 동일하게 동작

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class GuGu extends JFrame implements ActionListener {
    JTextField dan;
    JTextArea gugu;
    JLabel jl;
    JPanel pl;

    public GuGu() {
        setTitle("구구단");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```



Thank You!