



JAVA

GU, DA HAE

1. 자

2. 변

3. 연

4. 제

5. 배

6. 객

7. 예

8. 제

산

어

외

네

바

수

자

문

열

체

리

릭

JAVA

I N D E X



JAVA

절 차 지 향
객 체 지 향
객 체 지 향 특 징

객 체 지 향

J A V A

절차 지향 프로그래밍 Procedural Programming

절차지향 프로그래밍은 **절차**를 중심으로 알고리즘(문제 해결 방법)을 기술하겠다는 뜻이다.

현실의 문제는 다양한 변수를 지니고 있기 때문에 같은 문제라도 **다른 절차대로 해결해야 할 경우**가 많다. 때문에 절차 지향 프로그래밍을 대신할 개발 기법들이 많이 나왔다.

함수형 프로그래밍

명령형 프로그래밍

선언형 프로그래밍

구조적 프로그래밍

객체 지향 프로그래밍

...

c 가

>> ,

ex) .

1. 550cc .

but ,

2. 1 30

but ,

가

C++, Java, Python

객체 지향 프로그래밍

Object Oriented Programming; OOP

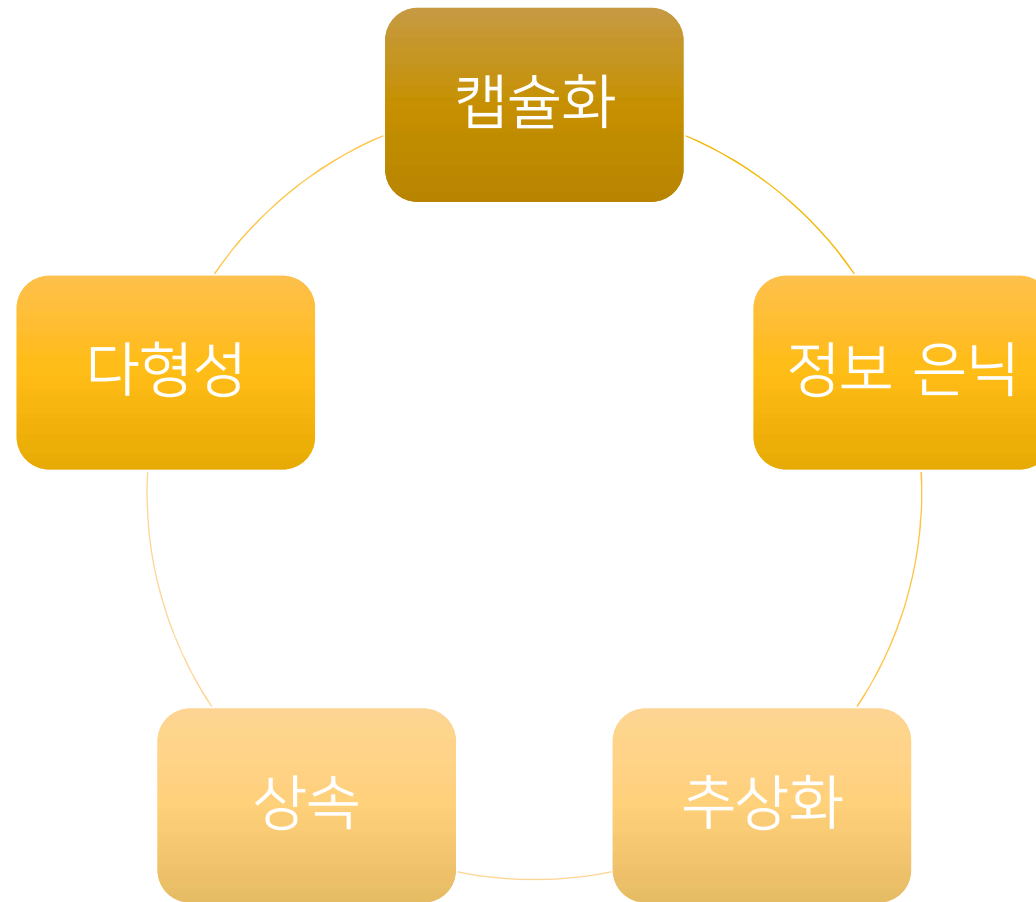
OOP는 절차가 아닌 **데이터**를 중심으로 알고리즘을 기술하는 기법이다.

알고리즘에서 필요한 **데이터를 먼저 정의**하고 **데이터에 절차를 결합**하여 '**객체**'를 만든다. 이렇게 만들어진 객체를 조립하여 프로그램을 완성한다.

```
[TV      ]  
:        ,  
:
```

JAVA

OOP 특징



캡슐화

Encapsulation

자료와 동작(기능)을 하나로 묶는 것. 이렇게 묶어 놓은 것을 객체라고 한다.

J A V A

J A V A

정보 은닉

Information Hiding

객체는 자료와 자료를 관리하는 동작으로 구성한다. 이 중 외부에서 꼭 필요한 동작(기능)만 공개하고 나머지는 **숨긴다**. 외부에서 객체의 자료를 마음대로 바꾸거나 허용하지 않는 동작을 하지 못하게 함으로써 **'안정성'을 확보**한다.

추상화

Abstraction

현실에 존재하는 사물을 객체로 표현하기 위해 사물의 **특징과 동작을 정의**해야 하는데 이를 '**데이터 모델링**'이라 한다. 모델링으로 필요한 자료와 동작이 결정되면 이들을 캡슐화하여 객체로 정의한다.

J A V A

상속

Inheritance

이미 만들어진 클래스를 기반으로 새로운 클래스를 정의하는 방법이다. 이 때 상속하는 클래스를 '상위 클래스', 상속 받는 클래스를 '하위 클래스'라 부른다.

J A V A

다형성

Polymorphism

똑같은 호출이라도 상황에 따라, 호출하는 객체에 따라 다른 동작을 할 수 있는 능력을 다형성이라 한다.

J D K
I D E
실 행 원 리
표 준 출 력

JAVA

자바

1. JDK 설치

| Java Development Kit

자바

ORACLE

Menu

Sign In

Country/Region

Call

Oracle Technology Network / Java

Software Downloads [View All Downloads](#)

Top Downloads

- Java SE
- Java EE and GlassFish
- JavaFX
- Java ME
- JDeveloper and ADF
- Enterprise Pack for Eclipse
- NetBeans IDE

New Downloads

- Java SE 10
Released 2018/03/20
- Java SE 9.0.4
Released 2018/01/16
- Java SE 8 Update 161/ 162
Released 2018/01/16
- Java SE Embedded 8 Update 161
Released 2018/01/16
- * Java CPE and PSII

자바로 프로그래밍을 하려면 가장 먼저 JDK가 필요하다.

<http://www.oracle.com/technetwork/java/index.html> 로 접속해보자.

Top Downloads의 **JAVA SE**를 선택하자.



자바

J A V A



Oracle Technology Network / Java / Java SE / Downloads

Overview Downloads Documentation Community Technologies Training

Java SE Downloads

 **DOWNLOAD** 

Java Platform (JDK) 10

 **DOWNLOAD** 

NetBeans with JDK 8

Java SDKs and Tools

- [Java SE](#)
- [Java EE and Glassfish](#)
- [Java ME](#)
- [Java Card](#)
- [NetBeans IDE](#)
- [Java Mission Control](#)

Java Resources

- [Java APIs](#)
- [Technical Articles](#)

Java Platform (JDK) 10 위의 **DOWNLOAD** 버튼을 누르자.

[Java Card](#)
[Java TV](#)
[New to Java](#)
[Community](#)
[Java Magazine](#)

language and running on the Java platform.

See also:

- [Java Developer Newsletter](#): From your Oracle account, select **Subscriptions**, expand **Technology**, and subscribe to **Java**.
- [Java Developer Day](#) hands-on workshops (free) and other events
- [Java Magazine](#)

[JDK 10 checksum](#)

Java SE Development Kit 10

You must accept the [Oracle Binary Code License Agreement for Java SE](#) to download this software.

☒ [Accept License Agreement](#)
☐ [Decline License Agreement](#)

Product / File Description	File Size	Download
Linux	305.93 MB	jdk-10_linux-x64_bin.rpm
Linux	338.37 MB	jdk-10_linux-x64_bin.tar.gz
macOS	395.42 MB	jdk-10_osx-x64_bin.dmg
Solaris SPARC	206.77 MB	jdk-10_solaris-sparcv9_bin.tar.gz
Windows	390.08 MB	jdk-10_windows-x64_bin.exe

Java Resources

- [Java APIs](#)
- [Technical Articles](#)
- [Demos and Videos](#)
- [Forums](#)
- [Java Magazine](#)
- [Developer Training](#)
- [Tutorials](#)
- [Java.com](#)

Accept License Agreement를 눌러 **라이선스 동의**하자. 하지 않으면 내려 받을 수 없다.

[jdk-10_windows-x64_bin.exe](#)를 눌러 설치 파일을 다운 받자.

J A V A

자바



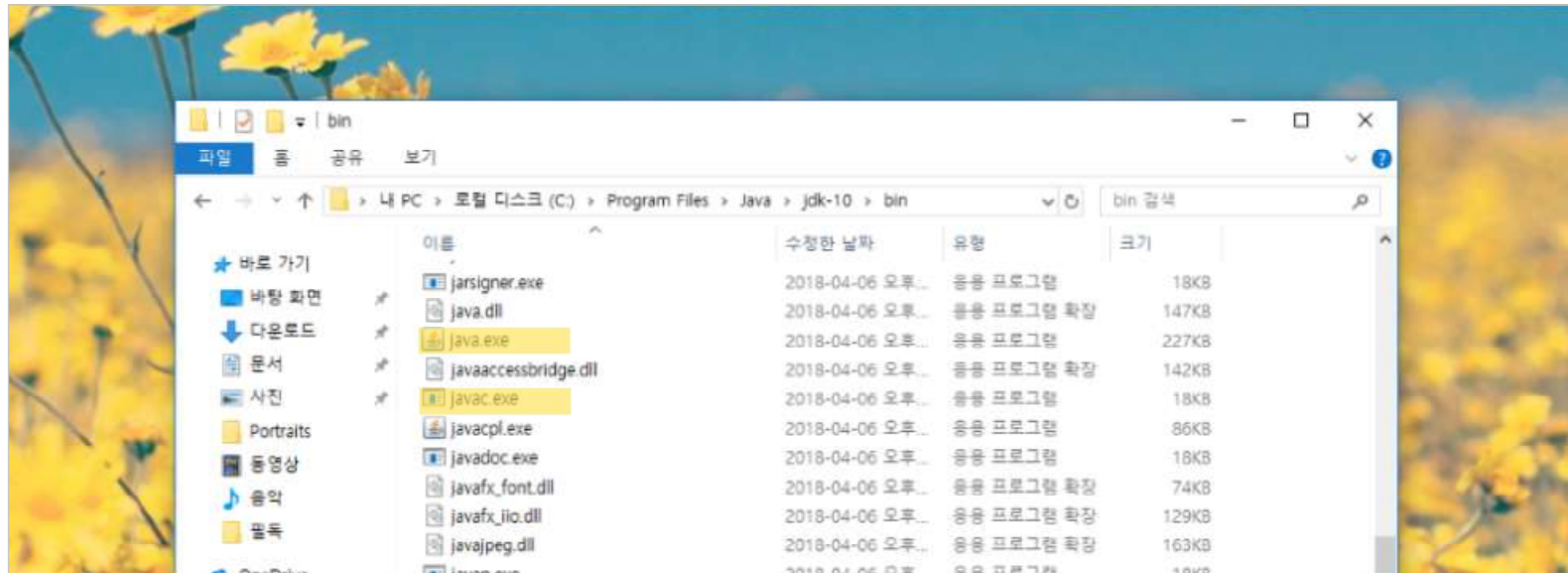
계속 **Next**를 누르자.

설치가 완료 되었다면 **Finish**를 선택하자.

2. 설치 확인

| 당신은 과연 설치를 제대로 했을까?

자바

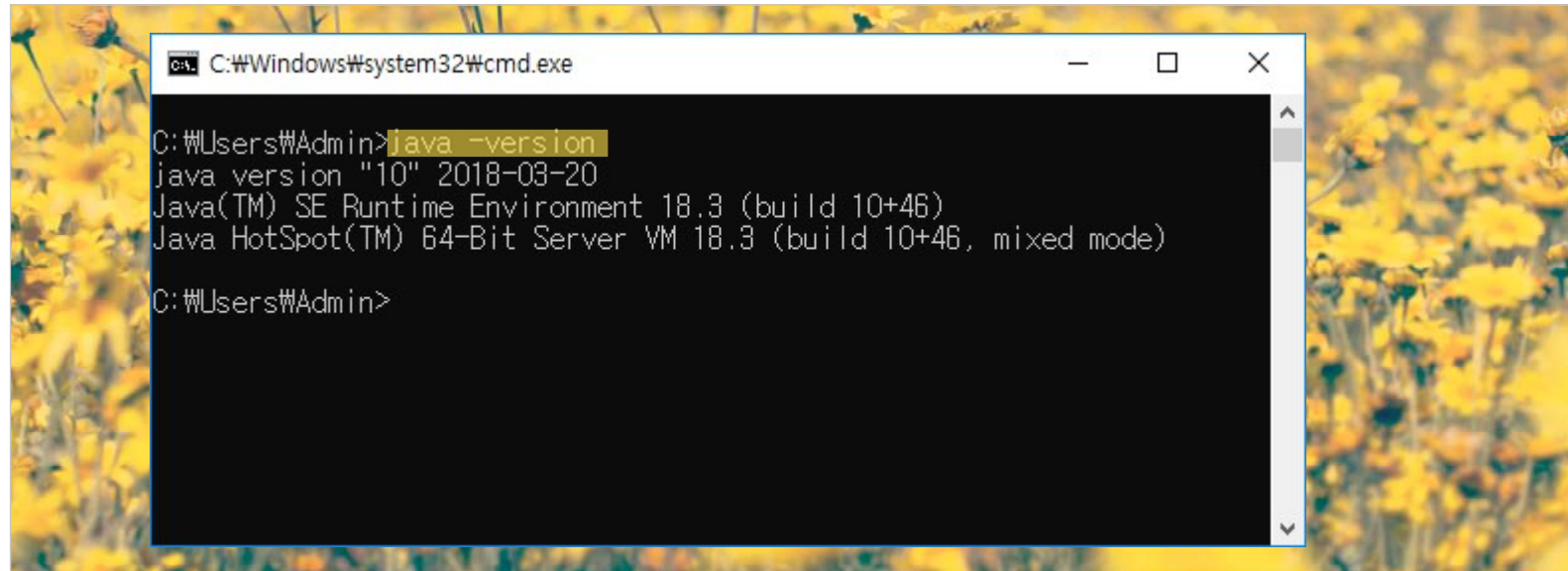


C:\WProgram Files\Java로 접근해 보자. java 안에는 **jdk**와 **jre** 디렉터리가 있다. 디렉터리 뒤의 숫자는 설치한 버전을 뜻한다. 다시 jdk의 bin 디렉터리로 이동해보자. 아주 중요한 두 실행파일을 볼 수 있을 것이다.

javac.exe	자바 컴파일러(Java Compiler)
java.exe	자바 런처(Launcher)

J A V A

자바

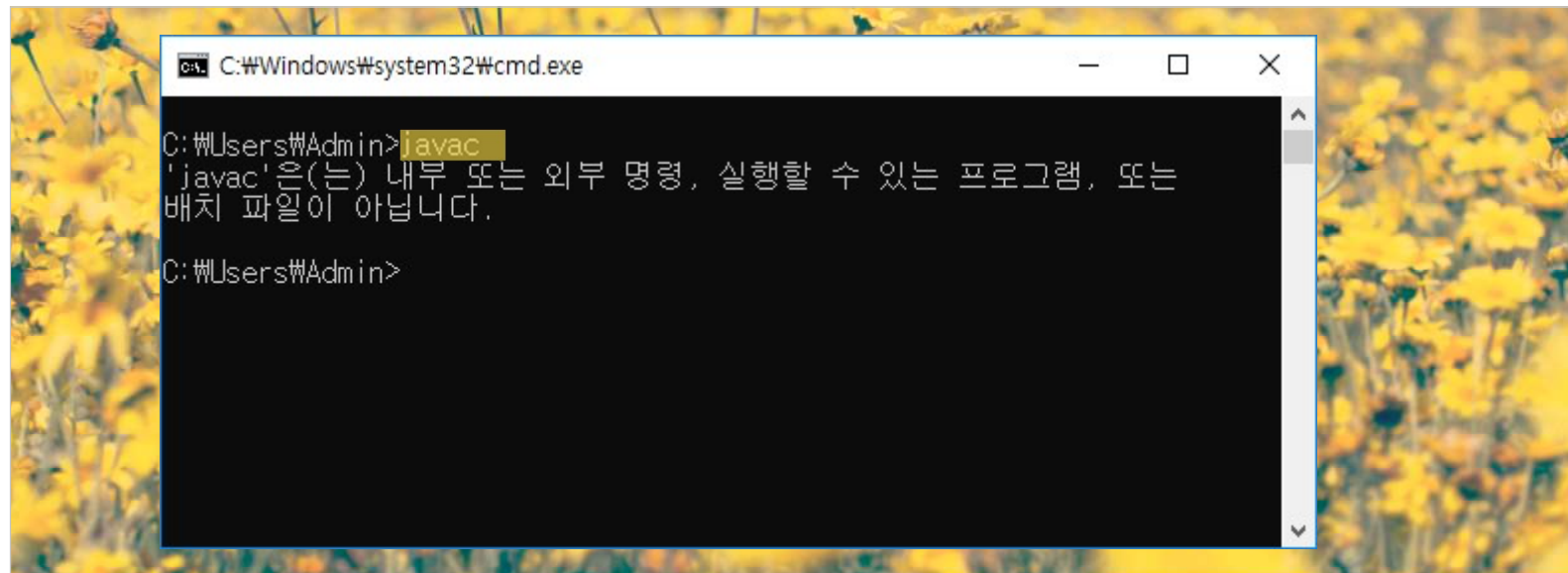
A screenshot of a Windows command prompt window. The title bar reads 'C:\Windows\system32\cmd.exe'. The command prompt shows the user 'Admin' at 'C:\Users\Admin>'. The command 'java -version' has been entered and executed. The output is: 'java version "10" 2018-03-20', 'Java(TM) SE Runtime Environment 18.3 (build 10+46)', and 'Java HotSpot(TM) 64-Bit Server VM 18.3 (build 10+46, mixed mode)'. The prompt is now 'C:\Users\Admin>'. The background of the window is a blurred image of yellow flowers.

```
C:\Windows\system32\cmd.exe
C:\Users\Admin>java -version
java version "10" 2018-03-20
Java(TM) SE Runtime Environment 18.3 (build 10+46)
Java HotSpot(TM) 64-Bit Server VM 18.3 (build 10+46, mixed mode)
C:\Users\Admin>
```

명령 프롬프트(cmd)를 실행해서 `java -version`을 입력해보자. 위와 같이 뜬다면 java 런처가 잘 설치되었다는 뜻이다.

J A V A

자바

A screenshot of a Windows Command Prompt window titled "C:\Windows\system32\cmd.exe". The window shows the command prompt "C:\Users\Admin>" followed by the command "javac". The output is a Korean error message: "'javac'은(는) 내부 또는 외부 명령, 실행할 수 있는 프로그램, 또는 배치 파일이 아닙니다." (The command 'javac' is not an internal or external command, a program that can be run, or a batch file). The prompt then returns to "C:\Users\Admin>". The background of the window is a blurred image of yellow flowers.

```
C:\Windows\system32\cmd.exe
C:\Users\Admin>javac
'javac'은(는) 내부 또는 외부 명령, 실행할 수 있는 프로그램, 또는
배치 파일이 아닙니다.
C:\Users\Admin>
```

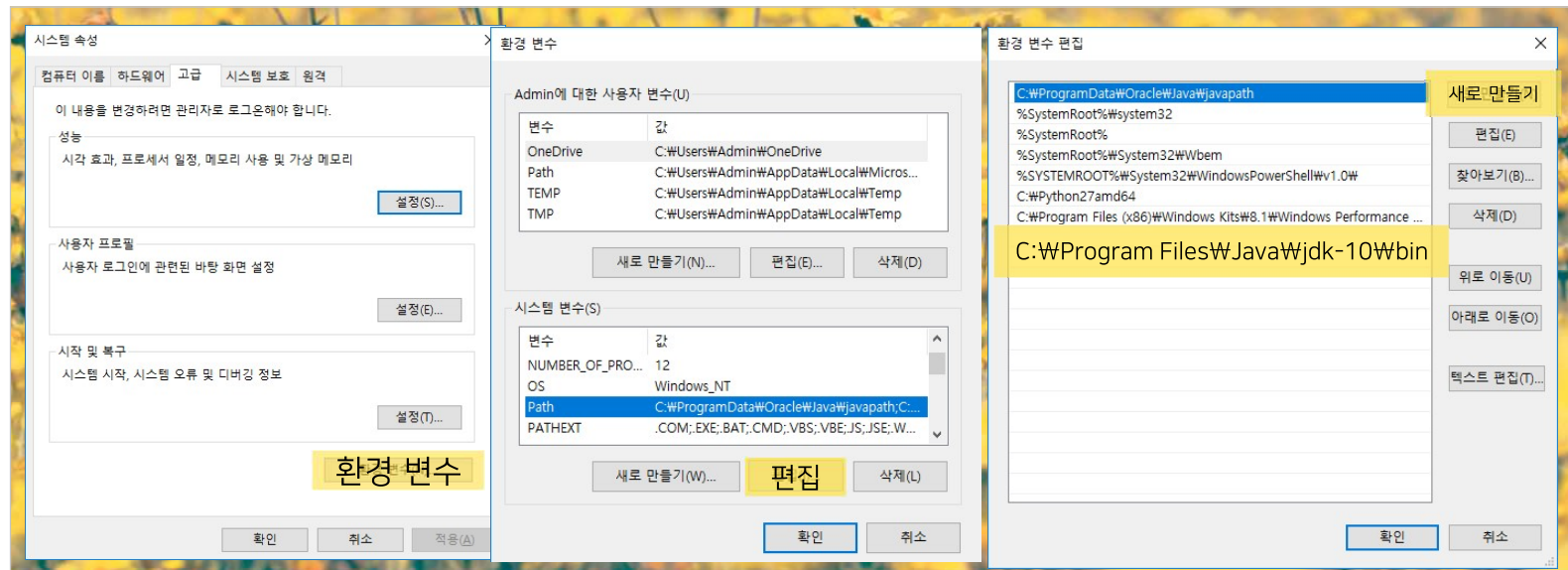
명령 프롬프트(cmd)를 실행해서 `javac`를 입력해보자. 위와 같이 뜬다면 java 컴파일러를 사용 할 수 없다.

이 경우에는 2-1번을 진행하자.

2-1. 환경변수

| java 컴파일러를 사용할 수 없다면 따라하자

자바



시스템 속성 > 고급 > 환경 변수를 눌러 환경 변수 창을 열자.

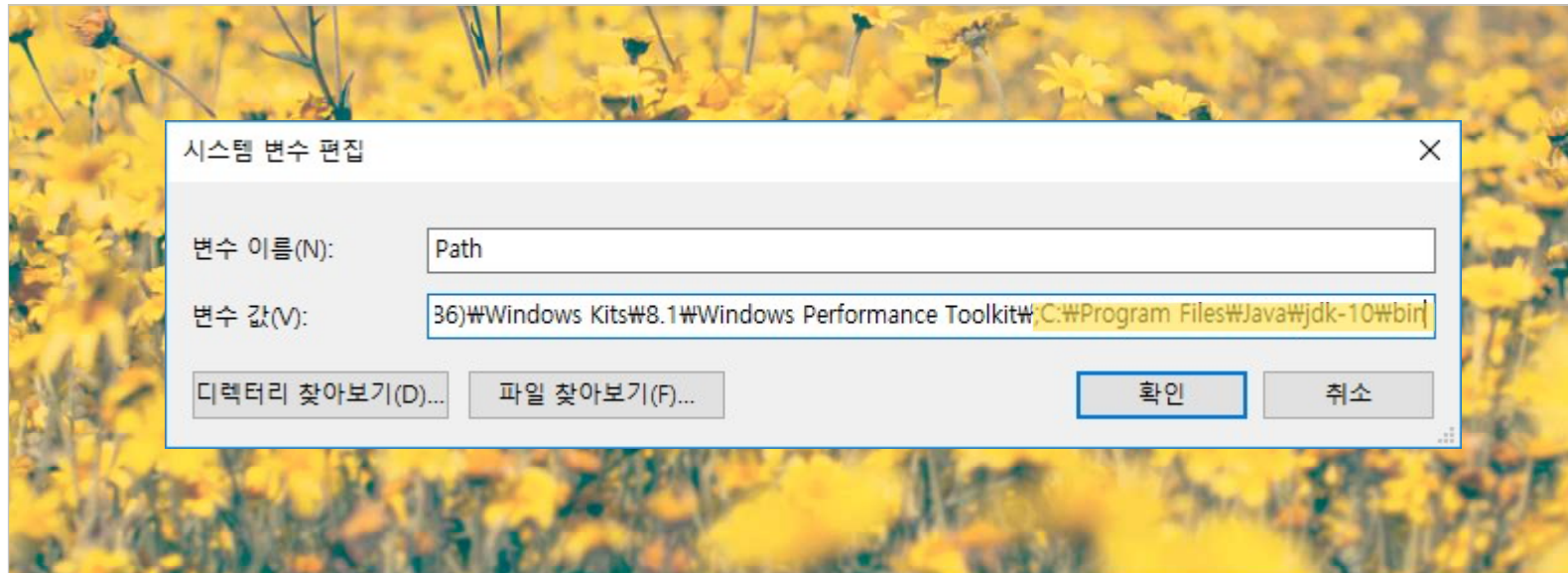
시스템 변수의 Path를 선택하고 편집을 누르자.

새로 만들기를 선택하여 jdk 디렉터리의 bin 디렉터리 경로를 집어넣자.

위의 과정을 끝냈다면 확인을 누르며 창을 하나씩 닫자.

2-1-1. 시스템 변수 편집 창이 다르다

자바



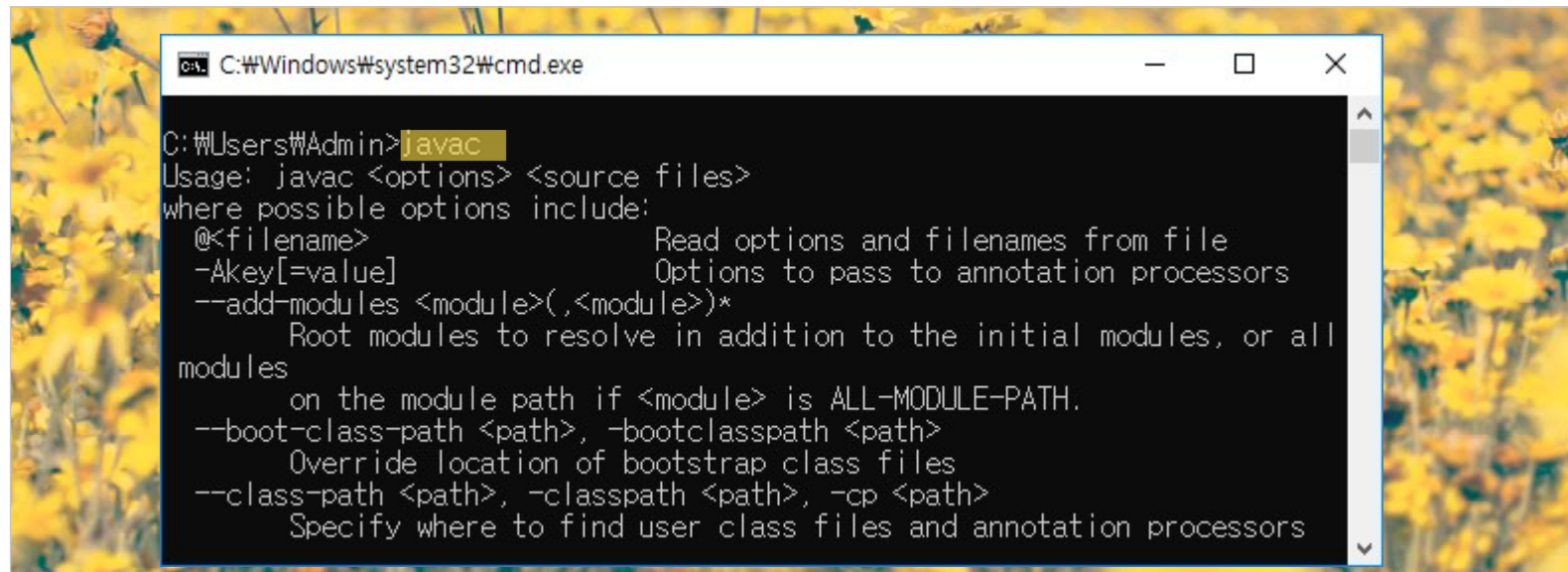
시스템 변수 편집 창이 앞과 다르다면 이렇게 하자.

우선 변수 값 가장 뒤에 **;(세미콜론)**을 넣자. 이미 세미콜론이 있다면 넣지 말자.

그 다음 **bin 디렉터리**의 **경로**를 넣고 확인을 누르자.

J A V A

자바

A screenshot of a Windows command prompt window titled "C:\Windows\system32\cmd.exe". The prompt shows the command "javac" being entered. The output displays the usage of the javac command, including options like -Akey, --add-modules, --boot-class-path, and --class-path.

```
C:\Windows\system32\cmd.exe
C:\Users\Admin>javac
Usage: javac <options> <source files>
where possible options include:
  @<filename>                Read options and filenames from file
  -Akey[=value]              Options to pass to annotation processors
  --add-modules <module>(,<module>)*
                             Root modules to resolve in addition to the initial modules, or all
                             modules
                             on the module path if <module> is ALL-MODULE-PATH.
  --boot-class-path <path>, -bootclasspath <path>
                             Override location of bootstrap class files
  --class-path <path>, -classpath <path>, -cp <path>
                             Specify where to find user class files and annotation processors
```

명령 프롬프트(cmd)를 실행해서 **javac**를 다시 입력해보자. 이번엔 성공이다.

J D K
I D E
실 행 원 리
표 준 출 력

JAVA

자바

IDE 설치

설치 PDF 참조

```
SE -  
EE -      +  
ME -  
      :      (ex.가      )  
SE  EE
```

J D K
I D E
실 행 원 리
표 준 출 력

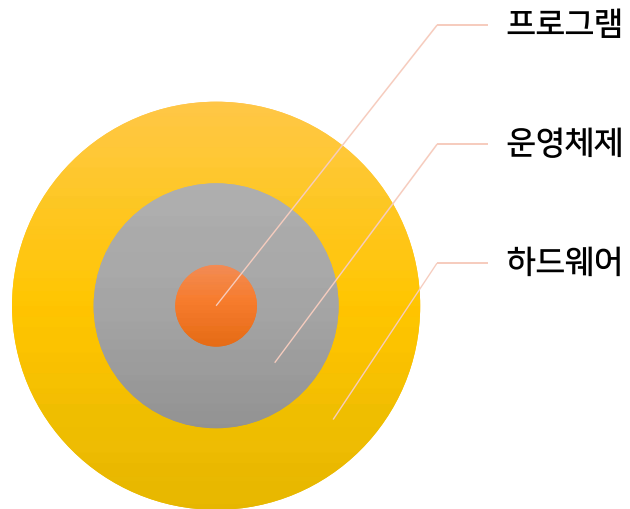
JAVA

자바

프로그램 실행 원리

C의 경우 전처리, 컴파일, 링크 과정(통틀어 빌딩이라 한다)을 거쳐 완성된 실행파일(프로그램)은 Windows나 Linux 혹은 Android 같은 운영체제 위에서 실행된다.

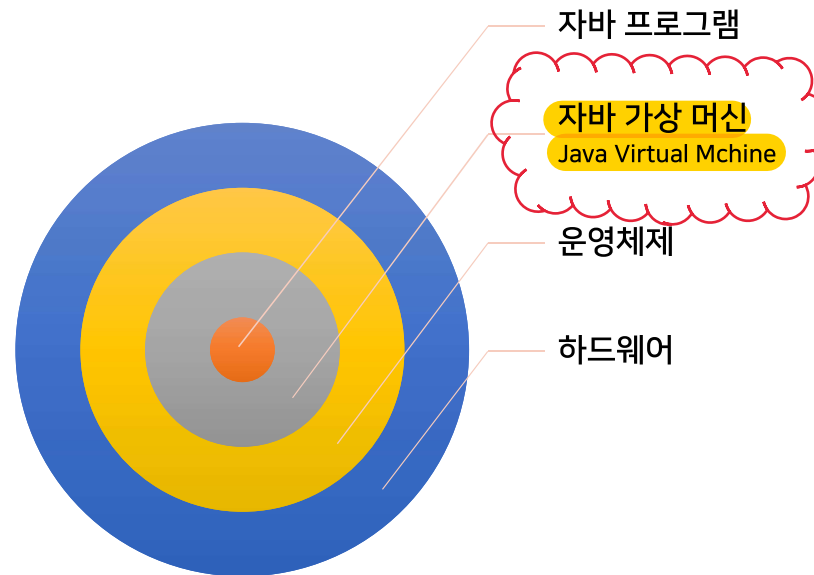
J A V A



J A V A

자바 프로그램 실행 원리

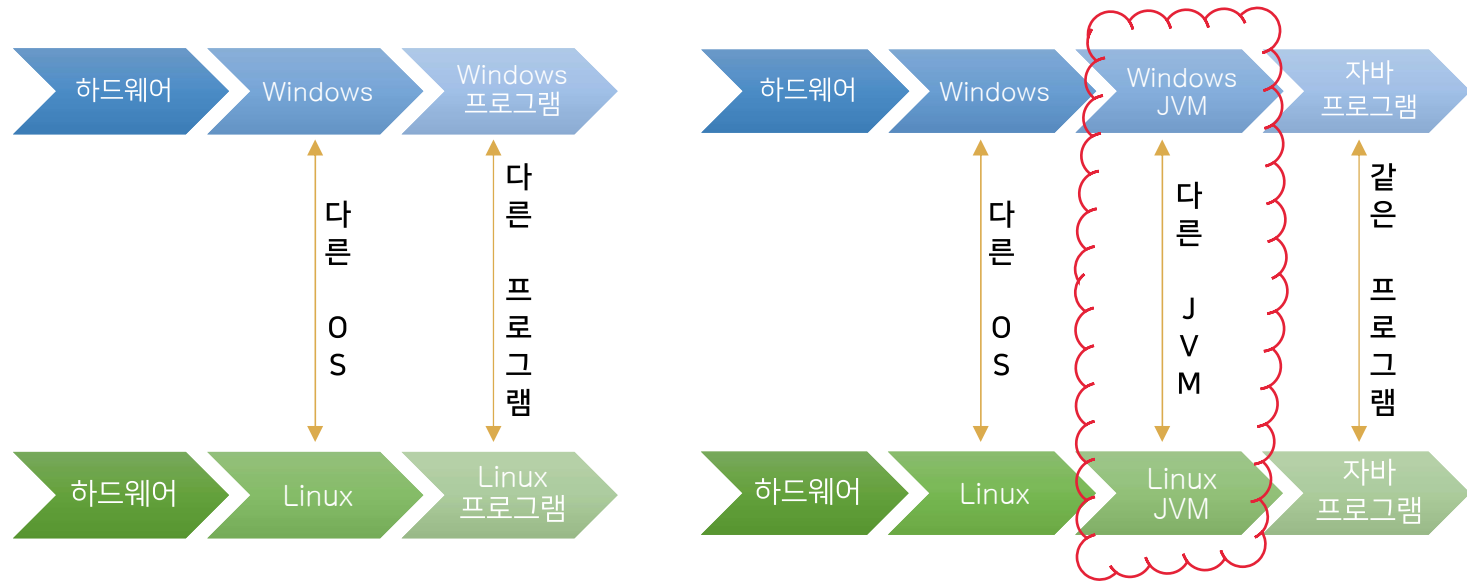
자바 프로그램과 운영체제 사이에 **자바 가상 머신JVM**이 끼어들었다. 운영체제가 JVM을 실행하면 JVM이 자바 프로그램을 실행시키는 구조다.



J A V A

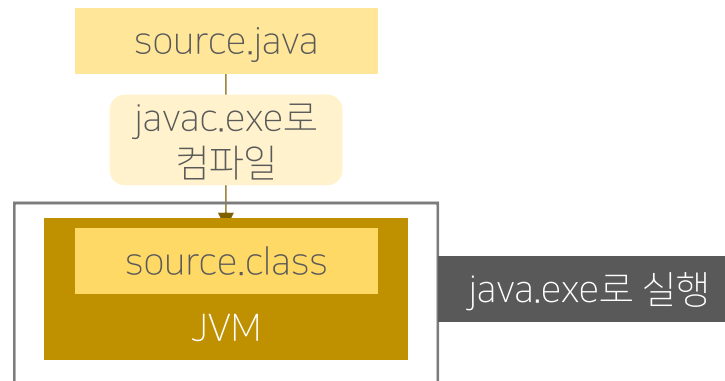
왜?

운영체제에 상관없이 프로그램을 동작 시키기 위함이다.



javac와 java의 역할

C 컴파일러 처럼 java 컴파일러도 소스 코드를 번역하는 프로그램이다. C 컴파일러가 C언어를 운영체제가 바로 이해할 수 있는 기계어(실행파일)로 번역했다면, java 컴파일러는 JVM이 이해할 수 있는 바이트코드(Bytecode, 클래스 파일)로 번역한다.



java 런처는 JVM을 실행시키고, 그 위에 자바 프로그램이 실행되도록 돕는 프로그램이다.

J D K
I D E
실 행 원 리
표 준 출 력

JAVA

자바

예제

```
public class Ex01{  
    public static void main(String[] args){  
        System.out.println("Hello");  
        System.out.println(7);  
        System.out.println(3.14);  
        System.out.println("3 + 4 = " + 7);  
        System.out.println(3 + 4);  
    }  
}
```

표준 출력

| `System.out.println`

앞의 예제는 하나의 클래스와 한 개의 메서드로 구성되어 있다. 이 예제로 알 수 있는 사실은 다음과 같다.

1. 프로그램을 실행하면 main 메서드 안의 명령문이 순차적으로 실행된다.
2. 클래스 이름이 Ex01이면 생성되는 클래스 파일의 이름이 Ex01이다.
3. `System.out.println`은 괄호 안에 출력하고 싶은 것을 넣으면 출력된다.
4. `System.out.println`은 출력한 뒤 개행을 한다.
5. `System.out.println`은 이어서 출력하고 할 때 + 기호로 출력 대상들을 연결하면 된다.
6. `System.out.println`에서 출력 대상이 전부 숫자일 때 + 기호는 연결이 아닌 덧셈을 한다.

문제

1. 다음 두 문장은 어떻게 실행될까?

```
System.out.println("2+5=" + 2 + 5);
```

```
System.out.println("2+5=" + (2 + 5));
```

2. 숫자 12를 5회 출력하는 프로그램을 작성해 보자. 단, 전부 같은 명령문이면 안된다.