Java chapter 2

연산자와 연산식
조건문(if문)
조건문(switch case break문)

2-1. 연산자와 연산식

연산자 : 연산에 사용되는 표시나 기호

피연산자 : 연산자와 함께 연산되는 데이터

연산식 : 연산자와 피연산자를 사용하여 연산 과정을 기술한 것

```
x + y
  ↑
  └── 연산자
x - y
  ↑ ↑
  └─┴── 피연산자
x * y + z

x == y
```

연산자 종류	연산자	피연산자 수	산출값	기능
산술	+, -, *, / , %	이항	숫자	사직연산 및 나머지 계산
부호	+, -	단항	숫자	음수와 양수의 부호
문자열	+	이항	문자열	두 문자열을 연결
대입	=, +=, -=, *=, /=, %=	이항	다양	우변의 값을 좌변의 변수에 대입
증감	++, --	단항	숫자	1만큼 증가/감소
비교	==, !=, >, <, <=, >=	이항	boolean	값의 비교
논리	!, &, |, &&, ||	단항, 이항	boolean	논리 부정, 논리곱, 논리합
조건	조건식 ? A : B	삼항	다양	조건에 따라 A 혹은 B

연산자	연산 방향	우선순위
증감(++, --), 부호(+, -), 논리(!)	←	높음
산술(*, /, %)	→	
산술(+, -)	→	
비교(<, >, <=, >=, instanceof)	→	
비교(==, !=)	→	
논리(&)	→	
논리(^)	→	
논리(|)	→	
논리(&&)	→	
논리(||)	→	
조건(?:)	→	
대입(=, +=, -=, *=, /=, %=)	←	낮음

- 모든 연산자 중 '=' 연산자는 항상 마지막에 연산한다.
- '&&', '||' 중 우선순위가 높은 것은 '&&' 이다.
- 연산자의 우선순위가 해깔린다면 괄호를 적극적으로 사용한다.

산술 연산자

연산식			예	설명
피연산자	+	피연산자	7 + 3	덧셈 연산
피연산자	-	피연산자	7 – 3	뺄샘 연산
피연산자	*	피연산자	7 * 3	곱셈 연산
피연산자	/	피연산자	7 / 3	나눗셈 연산
피연산자	%	피연산자	7 % 3	나머지를 구하는 연산

비교연산자(비교 연산자의 실행 결과는 true, false의 boolean 값을 가짐

구분	연산식			예	설명
동등 비교	피연산자1	==	피연산자2	3 == 5	두 피연산자의 값이 같은지 검사
	피연산자1	!=	피연산자2	3 != 5	두 피연산자의 값이 다른지 검사
크기 비교	피연산자1	>	피연산자2	3 > 5	피연산자1이 큰지를 검사
	피연산자1	>=	피연산자2	3 >= 5	피연산자1이 크거나 같은지 검사
	피연산자1	<	피연산자2	3 < 5	피연산자1이 작은지를 검사
	피연산자1	<=	피연산자2	3 <= 5	피연산자1이 작거나 같은지를 검사

논리부정 연산자 (true 값은 false로, flase 값은 true로 변경)

연산식		예	설명
!	피연산자	!true	피연산자가 true이면 false 값을 산출 피연산자가 false이면 true 값을 산출

ex> boolean b = !true;

System.out.println(b); //false

증감연산자

연산식		예	설명
++	피연산자	++5	단독으로 사용하면 1증가, 같은 줄에 다른 코드와 함께 사용되면 1증가 후 다른 코드 실행
--	피연산자	--5	단독으로 사용하면 1감소, 같은 줄에 다른 코드와 함께 사용되면 1감소 후 다른 코드 실행
피연산자	++	5++	단독으로 사용하면 1증가, 같은 줄에 다른 코드와 함께 사용되면 다른 코드 실행 후 1증가
피연산자	--	5--	단독으로 사용하면 1감소, 같은 줄에 다른 코드와 함께 사용되면 다른 코드 실행 후 1감소

사용예시

1) int x = 10;
++x;

2) int x = 10;
x--;

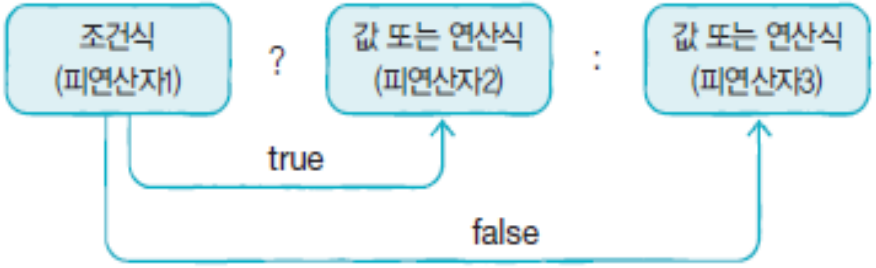
3) int x = 10;
int y = ++x;

4) int x = 10;
int y = x--;

복합 대입 연산자 (산출 연산자와 대입 연산자(=)를 간결하게 사용하는 연산자)

- a += b; -> a = a + b; ex> a += 3; -> a = a + 3;
- a -= b; -> a = a - b; ex> a -= 1; -> a = a - 1;
- a *= b; -> a = a * b; ex> a *= 2; -> a = a * 2;
- a /= b; -> a = a / b; ex> a /= 5; -> a = a / 5;
- a%= b; -> a = a % b; ex> a %= 3; -> a = a % 3;

삼항연산자



사용예시

```
int a = 10 > 3 ? 5 : 1;
System.out.println(3 < 5 ? "참" : "거짓");
```

2-2. 조건문(if문)

조건문이란 특정 조건을 만족할 때만 코드를 실행하는 문법을 말한다. 자바의 조건문에는 if문과 switch case break문 2가지가 있다.

이번 시간에는 if문에 대해 살펴보자.

if문 기본문법

```
if(참 또는 거짓을 판별할 수 있는 조건){
    실행내용;
    실행내용;
    ...
}
```

- if문의 소괄호 안 조건이 참일 경우에만 if문 중괄호 안의 내용을 실행!
- 중괄호 안에는 여러줄의 실행문이 올 수 있음
- 중괄호 안의 실행문이 하나일 경우는 중괄호 생략 가능
- if문은 '만약에...라면' 으로 해석한다.

```java
public static void main(String[] args) {
    System.out.println("프로그램 시작");

    int x = 3;
    if(x > 1){
        System.out.println("A");
        System.out.println("B");
    }

    System.out.println("프로그램 종료");
}
```

```
프로그램 시작
A
B
프로그램 종료
```

```java
public static void main(String[] args) {
    System.out.println("프로그램 시작");

    int x = 3;
    if(x > 5){
        System.out.println("A");
        System.out.println("B");
    }

    System.out.println("프로그램 종료");
}
```

```
프로그램 시작
프로그램 종료
```

```java
public static void main(String[] args) {
    System.out.println("프로그램 시작");

    int x = 3;
    if(x > 5)
        System.out.println("A");

    System.out.println("프로그램 종료");
}
```

```
프로그램 시작
프로그램 종료
```

중괄호 안의 실행문이 하나일 경우는 중괄호 생략 가능

```java
public static void main(String[] args) {
    System.out.println("프로그램 시작");

    int x = 3;

    if(x > 1) {
        System.out.println("A");
    }

    if(x > 2){
        System.out.println("B");
    }

    System.out.println("프로그램 종료");
}
```

```
프로그램 시작
A
B
프로그램 종료
```

if문은 필요하다면 여러번 사용할 수 있음

if문은 else문과 함께 사용할 수 있다. else는 '그 밖에는...' 또는 '그렇지 않으면...'으로 해석한다.

if-else문 문법

```
if(참 또는 거짓을 판별할 수 있는 조건){
    실행내용;
    실행내용;
    ...
}
else{
    실행내용;
    실행내용;
    ...
}
```

- else문은 if문 없이 단독으로 사용할 수 없다.
- else문은 if문과 다르게 소괄호 부분이 없다.
- if-else문은 if와 else문 하나만 반드시 실행된다.

```java
public static void main(String[] args) {
    System.out.println("프로그램 시작");

    int x = 3;

    if(x > 1) { //x가 1보다 크면...
        System.out.println("A");
    }

    else{ //그렇지 않으면...
        System.out.println("B");
    }

    System.out.println("프로그램 종료");
}
```

```java
public static void main(String[] args) {
    System.out.println("프로그램 시작");

    int x = 3;

    if(x > 1)  //x가 1보다 크면...
        System.out.println("A");
    else //그렇지 않으면...
        System.out.println("B");

    System.out.println("프로그램 종료");
}
```

```
프로그램 시작
A
프로그램 종료
```

- 위 코드는 “A”, “B”중 무조건 하나만 출력한다.
- else문도 if문과 마찬자기로 실행문이 하나라면 중괄호를 생략할 수 있다.

if문은 else if, else문과 함께 사용할 수 있다. else if는 '그렇지 않고 만약에...'로 해석한다.

if – else if – else문 문법

```
if(참 또는 거짓을 판별할 수 있는 조건){
    실행내용;
    실행내용;
    ...
}
else if(참 또는 거짓을 판변할 수 있는 조건){
    실행내용;
    실행내용;
    ...
}
else{
    실행내용;
    ...
}
```

- else if문은 if문 없이 단독으로 사용할 수 없다.
- else if는 필요하다면 여러개 사용할 수 있다.
- else if와 else문이 함께 사용된다면 반드시 else문이 마지막에 온다.
- if, else if, else문 중 무조건 하나만 반드시 실행된다.
- else if문의 실행 내용이 하나라면 중괄호를 생략할 수 있다.

```java
public static void main(String[] args) {
    System.out.println("프로그램 시작");

    int x = 3;

    if(x > 1) {  //x가 1보다 크면...
        System.out.println("A");
    }
    else if(x > 5) { //그렇지 않고 x가 5보다 크면...
        System.out.println("B");
    }
    else{ //그렇지 않으면...
        System.out.println("C");
    }

    System.out.println("프로그램 종료");
}
```

```
프로그램 시작
A
프로그램 종료
```

```java
public static void main(String[] args) {
    System.out.println("프로그램 시작");

    int x = 3;

    if(x > 1) {  //x가 1보다 크면...
        System.out.println("A");
    }
    else if(x > 5) { //그렇지 않고 x가 5보다 크면...
        System.out.println("B");
    }
    else if(x > 7){ //그렇지 않고 x가 7보다 크면...
        System.out.println("C");
    }
    else{ //그렇지 않으면...
        System.out.println("D");
    }

    System.out.println("프로그램 종료");
}
```

```
프로그램 시작
A
프로그램 종료
```

- else if는 여러번 가용 가능!
- else문과 esle if문이 함께 사용되면 else문이 마지막이 된다.

```java
public static void main(String[] args) {
    System.out.println("프로그램 시작");

    int x = 3;

    if(x > 1) {  //x가 1보다 크면...
        System.out.println("A");
    }
    else if(x > 5) { //그렇지 않고 x가 5보다 크면...
        System.out.println("B");
    }
    else if(x > 7){ //그렇지 않고 x가 7보다 크면...
        System.out.println("C");
    }

    System.out.println("프로그램 종료");
}
```

```
프로그램 시작
A
프로그램 종료
```

```java
public static void main(String[] args) {
    System.out.println("프로그램 시작");

    int x = 0;

    if(x > 1) {  //x가 1보다 크면...
        System.out.println("A");
    }
    else if(x > 5) { //그렇지 않고 x가 5보다 크면...
        System.out.println("B");
    }
    else if(x > 7){ //그렇지 않고 x가 7보다 크면...
        System.out.println("C");
    }

    System.out.println("프로그램 종료");
}
```

```
프로그램 시작
프로그램 종료
```

- 위와 같이 else if문을 작성한다고 무조건 else문이 있는건 아니다.
- 단, else 문이 없다면 if와 else if문의 내용 모두 경우에 따라 실행되지 않을 수 있다.
- 그렇다고 else문이 없을 경우, if문과 else if문 모두가 실행되는 일은 없다.

다음과 같은 if문이 있을 때 실행 결과를 예측해보자.

```java
public static void main(String[] args) {
    System.out.println("프로그램 시작");

    int x = 5;

    if(x > 3) {
        System.out.println("A");
    }
    else if(x > 4) {
        System.out.println("B");
    }
    else if(x == 5){
        System.out.println("C");
    }
}
```

```java
public static void main(String[] args) {
    System.out.println("프로그램 시작");

    int x = 5;

    if(x > 8) {
        System.out.println("A");
    }
    else if(x % 2 == 1) {
        System.out.println("B");
    }
    else if(x < 8){
        System.out.println("C");
    }
}
```

- if, else-if, else문은 하나의 쌍이기 때문에 위에서부터 해석하여 조건에 부합하는 조건이 발견되는 즉시 다음 조건은 해석조차 하지 않는다.
- 그렇기 때문에 여러 조건이 참이라 하더라도 최초의 참인 조건만 실행한다.

다음과 같은 if문이 있을 때 실행 결과를 예측해보자.

```
public static void main(String[] args) {
  System.out.println("프로그램 시작");

  int x = 5;

  if(x > 1)
    System.out.println("A");
  else if(x != 3)
    System.out.println("B");

  if(x == 5)
    System.out.println("C");
  else
    System.out.println("D");
}
```

'==' 연산자는 값이 같을 경우 ture이다. 이때 주의 사항이 있다. 문자열 데이터는 '==' 연산자로 비교할 수 없다!!

```
public static void main(String[] args) {
  String name = "홀길동";

  //잘못된 코드!!. 문자열은 '=='로 비교 불가능!
  if(name == "홍길동"){
    System.out.println("A");
  }
}
```

```
public static void main(String[] args) {
  String name = "홀길동";

  //문자열 비교는 equals() 명령어를 사용한다
  if( name.equals("홍길동") ){
    System.out.println("A");
  }
}
```

- 문자열 비교를 위해서는 equals() 명령어를 사용한다.
- 문자열A.equals(문자열B) -> A, B 문자열이 같으면 true를 반환한다.

```
System.out.println( "java".equals("java") );
System.out.println( "java".equals("Java") );

String str1 = "java";
System.out.println( str1.equals("java") );
String str2 = "python";
System.out.println( str1.equals(str2) );
```

```
true
false
true
false
```

2-3. 조건문(switch case break)

자바의 두번째 조건문 switch case break문을 알아보겠다. if문은 조건이 일치하는 부문만 실행하는 조건문이었다면 swtich case문은 조건이 맞는 부분부터 실행하는 문법이다.

switch case문 문법

```
switch(변수 또는 값){
    case 경우1 :
        실행문;
        실행문;
    case 경우2 :
        실행문;
        실행문;
    default :
        실행문;
        실행문;
}
```

- switch문의 소괄호 안에는 변수 또는 값이 들어온다.
- case문은 switch문의 소괄호 안에 작성한 변수나 값이 가질 수 있는 값을 작성한다.
- case문 안에는 조건이 맞을 경우 실행할 코드를 작성한다.
- default는 일치하는 case가 없을 때 실행하는 부분이다. if문으로 따지만 else와 같은 역할을 한다.
- default문은 필요한 경우만 작성하면 된다.
- 조건에 부합하는 case부분만 실행하는 것이 아니다! 조건에 부합하는 case부분부터 모든 내용을 실행한다.

```java
public static void main(String[] args) {
  int x = 5;

  switch(x){
    case 3:
      System.out.println("A");
    case 5 :
      System.out.println("B");
      System.out.println("C");
    case 7 :
      System.out.println("D");
    default :
      System.out.println("E");
  }
}
```

```
B
C
D
E
```

```java
public static void main(String[] args) {
  String str = "java";

  switch(str){
    case "python":
      System.out.println("A");
    case "java" :
      System.out.println("B");
      System.out.println("C");
    case "c++" :
      System.out.println("D");
  }
}
```

```
B
C
D
```

switch case문은 이전 슬라이드의 예시에서 확인하였 듯, 조건이 맞는 부분부터 모든 내용을 실행한다. 하지만 코드 작성 시 이러한 흐름을 원하는 경우는 잘 없다. 조건에 맞는 case만 실행하고 싶은 경우가 대부분이다. 이를 위해 case문은 break와 함께 사용한다. break문을 만나면 switch문 해석을 종료한다.

```
public static void main(String[] args) {
  int x = 3;

  switch(x){
    case 1:
      System.out.println("A");
      break;
    case 3 :
      System.out.println("B");
      break;
    case 5 :
      System.out.println("C");
  }
}
```

B

if문과 switch case 문은 각각 언제 사용할까?

조건이 범위일 경우에는 주로 if문을 사용하고, 조건이 특정 값인 경우에는 switch case문을 사용한다.