제어문

▶ 제어문

프로그램 수행의 흐름을 결정(제어)하는 데 사용되는 문장

프로그램은 항상 위에서 부터 아래로 실행되는데 조건에 따라 원하는 순서로 실행되도록 흐름을 조정 함

✓ 제어문의 종류

조건문 : if문, switch문
반복문 : while문, do-while문, for문
분기문 : continue , break , return 조건문, 반복문을 도와준다

목차

Chap01.
조건문

▶ 조건문

조건식을 제시하고 조건식의 결과에 따라 다음 코드의 수행 여부를 결정하는 문장

✓ 조건문의 종류

세부적으로 보면 선택문

if문

```
if(조건식1) {
        수행될 문장;
} else if(조건식2) {
        수행될 문장;
} else if(조건식3) {
        수행될 문장;
} else {
        수행될 문장;
}
```

switch문

```
switch(조건식) {
case 값1:
        수행될 문장;
        break;
case 값2:
        수행될 문장;
        break;
default:
        수행될 문장;
}
```

▶ if문

조건이 참일 때 다음에 오는 하나의 명령을 실행

조건이 참일 때 실행 할 내용이 한 줄 이상일 경우 { } 로 영역지정

✓ if문 사용법

중괄호로 안묶어주면 밑에 한줄만 인식함!

```
if(조건식)
        명령;
```

조건식의 결과 값이 true면 명령 실행

false면 실행하지 않음

```
if(조건식){
        명령1;
        명령2;
}
```

조건식의 결과 값이 true면 명령1,2 실행

false면 실행하지 않음

▶ if문

✓ if문 예시

```
if(num > 0) {
        System.out.println("양수입니다.");
}
```

✓ 예시 실행 결과

```
num = 10 인 경우
------------
양수입니다.
```

```
num = -5 인 경우
------------
(출력결과 없음)
```

▶ if-else문

거짓일때의 별도의 if문 작성하기 귀찮다
else문을 쓴다

if 문의 조건식의 결과가 true 일 경우와 false일 경우 수행할 문장을 각각 지정
조건이 참일 때 if 다음 명령을 실행하고 거짓일 때 else 다음 명령을 실행

✓ if-else문 사용법

```
if(조건식) {
        명령1;
} else {
        명령2;
}
```

조건식의 결과 값이 true면 명령1 실행
false면 명령2 실행

주의할 점
1. else의 조건은 if옆에 있다. if 없이는 else 혼자 쓸 수 없다.
2. else 다음에 조건이 와선 안된다. else는 위의 조건이 거짓일 때 실행하는 것이기 때문.

▶ if-else문

✓ if~else문 예시

```
if(num % 2 == 0) {
        System.out.println("짝수");
} else {
        System.out.println("홀수");
}
```

✓ 예시 실행 결과

```
num = 10 인 경우
------------
짝수
```

```
num = 5 인 경우
------------
홀수
```

▶ if – else if - else 문

조건을 2개 이상 주고 각 조건에 따라 다른 내용을 실행

else if는 내가 필요한 만큼 사용 가능
if와 else는 1개씩만 사용 가능

✓ if-else if-else 문 사용법

참이 아니라면
한번 더 조건을 주고 싶다 => else if를 한번 더 적어준다
또 그 else if가 참이 아닐 때 조건 주고 싶으면 또 else if

```
if(조건식1) {
        명령1;
} else if(조건식2){
        명령2;
} else {
        명령3;
}
```

위에 있는 조건이 모두 참이 아닐 때 else를 쓴다
참이 아닐 때 쓸게 없다 -> else 안써도 된다

조건식1의 결과 값이 true면 명령1 실행

false면 조건식2 확인

조건식2의 결과 값이 true면 명령2 실행

false면 명령3 실행

* if는 true, false와 상관 없이 조건절 실행,
  if~else if~else는 조건문이 true면 이후 조건은 실행하지 않음

▶ if – else if - else 문

✓ if~else if~else문 예시

* 주의할 점
0은 짝수가 되기도 함

```
if(num > 0) {
        System.out.println("0보다 큰 수");
} else if (num < 0){
        System.out.println("0보다 작은 수");
} else {
         System.out.println("0과 같은 수");
}
```

✓ 예시 실행 결과

```
num = 10 인 경우
------------
0보다 큰 수
```

```
num = -5 인 경우
------------
0보다 작은 수
```

```
num = 0 인 경우
------------
0과 같은 수
```

▶ if – else if - else 문

✓ if~else if~else문 예시

```
if(month == 1 || month == 2 || month == 12) {
        season = "겨울";
} else if(month >= 3 && month <= 5) {
        season = "봄";
} else if(month >= 6 && month <= 8) {
        season = "여름";
} else if(month >= 9 && month <= 11) {
        season = "가을";
} else {
        season = "해당하는 계절이 없습니다.";
}
```

```
int month = scan.nextInt();
String season;
if(month ==1 || month ==2 || month == 12) {
season = "겨울";
} else if(month >= 3 && month <=5) {
season = "봄";
} else if(month >= 6 && month <=8) {
season = "여름";
} else if(month >=9 && month <=11) {
season = "가을";
} else {
season = "해당하는 계절이 없습니다.";
}
 System.out.println(season);
```

변수는 값이 들어가야 사용할 수 있다.
season에 대한 모든 경우의 수를 입력해주거나
"" 의미없는 공백 초기화 해주거나

▶ if문

✓ 중첩 if

어차피 if문 다음문장 실행할 뿐임
또 if필요하면 또 써도 된다

```
if (조건식1) {
    if (조건식2) {
        if (조건식3) {
            수행될 문장;
        } else if (조건식4) {
            수행될 문장;
        } else {
            수행될 문장;
        }
    } else {
        수행될 문장;
    }     조건식이 거짓일 때 조건
} else if (조건식5) {
    수행될 문장;
} else {  다 거짓일 때 실행
    수행될 문장;
}
```

✓ 중첩 if문 예시

```
if (month == 1 || month == 2 || month == 12) {
            season = "겨울";
            if(temperature <= -15) {
                        season += " 한파 경보";
            } else if(temperature <= -12) {
                        season += " 한파 주의보";
            }
} else if (month >= 3 && month <= 5) {
            season = "봄";
} else if (month >= 6 && month <= 8) {
            season = "여름";
            if(temperature >= 35) {
                        season += " 폭염 경보";
            } else if(temperature >= 33) {
                        season += " 폭염 주의보";
            }
} else if (month >= 9 && month <= 11) {
            season = "가을";
} else {
            season = "해당하는 계절이 없습니다.";
}
```

▶ if문

-. 놀이 공원 프로그램 만들기

==========출력==========

##놀이 공원 프로그램##
입장 하실 인원은 총 몇 명입니까? 3
어른은 몇 명입니까? (인원당 1만5천원) 1
아이는 몇 명입니까? (인원당 5천원) 2
지불하실 총 금액은 25000원 입니다

if문

KH정보교육원

-. 계산기 만들기(if문 사용)

==========출력==========

1~4 먼저 선택

계산기

1. 더하기
2. 빼기
3. 나누기
4. 곱하기

선택 :

▶ switch문

전환을 시키는 것

조건의 결과가 값 →

case가 됨

조건 값에 해당하는 경우를 찾아 선택하여 실행 하는 것

상수 값(정수 또는 문자, 문자열)의 결과를 가지는 조건을 입력 받아 동일

한 case 조건 이후 내용부터 모두 실행

조건식의 결과 값과 일치하는 case문으로 이동

default문은 일치하는 case문이 없을 때 수행(= else)

▶ switch문

✓ switch문 사용법

```
switch(조건식) {
case 조건값1 :
        명령1;
case 조건값2 :
        명령2;
case 조건값3 :
        명령3;
default :
}
```

조건식의 결과값과 조건값 중 일치하는

case 를 찾아 이후 내용 모두 실행

▶ switch문

✓ switch문 예시

```
Scanner scan = new Scanner(System.in);
System.out.print("정수를 입력하세요 : ");
int select = scan.nextInt();
switch(select)
{
case 1 : System.out.println("1 입력");
case 2 : System.out.println("2 입력");
case 3 : System.out.println("3 입력");
case 4 : System.out.println("4 입력");
default :System.out.println("기타입력");
}
```

switch문

✓ switch문 예시

switch문도 중첩가능

```
Scanner scan = new Scanner(System.in);
System.out.print("정수를 입력하세요 : ");
int select = scan.nextInt();
switch(select)
{
case 1 : System.out.println("1 입력");
         break;
case 2 : System.out.println("2 입력");
         break;
case 3 : System.out.println("3 입력");
         break;
case 4 : System.out.println("4 입력");
         break;
default :System.out.println("기타입력");
}
```

여러줄 쓸거면 break 전까지 여러줄 써주면 됨

if를 제외한 제어문 하나를 빠져나와주는 분기문

default 도 else와 마찬가지로 필요없으면 안넣어줘도 됨

▶ switch문

✓ switch문 예시

```
switch(num % 5) {
case 1:
        team = "1조";
        break;
case 2:
        team = "2조";
        break;
case 3:
        team = "3조";
        break;
case 4:
        team = "4조";
        break;
default:
        team = "다시";
}
```

▶ switch 문

-. 계산기 만들기(switch문 사용)

==========출력==========

계산기
1. 더하기
2. 빼기
3. 나누기
4. 곱하기
선택 :

Chap02.
반복문

▶ 반복문

문장들을 반복해서 여러 번 수행되게 할 때 사용하는 구문

조건을 만족하는 동안 명령을 반복하여 실행

구문 상에 반복되는 구간을 루프(LOOP) 라고 함

✓ 반복문의 종류

while문

```
[초기식;]
while(조건식) {
        수행될 문장;
        [증감식 or 분기문;]
}
```

for문

```
for(초기식; 조건식; 증감식)
{
        수행될 문장;
}
```

▶ while문

-초기식
-조건식
-증감식
을 넣어줘야 한다.

✓ while

조건식이 계속true이면 무한루프

[초기식;] 조건이 참인 동안에 안에 있는 내용을 반복시켜주겠다

```
while(조건식) {
        수행될 문장;
        [증감식 or 분기문];
}
```

조건식이 true일 때 문장 수행

문장 수행이 끝나면 조건식

다시 확인 후 true면 수행,

조건식이 false가 될 때까지 수행

조건식이 false가 되면 반복문 종료

* { } 안에 조건을 벗어나게 할 연산(증감식, 분기문) 필요

✓ while문 예시

while문의 조건으로 사용하기 전에
반드시 초기값이 들어가있어야 한다.

```
int i = 1;  // 초기식
while(i <= 10) {
    System.out.println(i + " 출력");
    i++;
}
```

증감식
변수가 변할 수 있게 해줘야 한다.

✓ 실행 결과

```
1 출력
2 출력
...
9 출력
10 출력
```

▶ do-while문

✓ do ~ while

```
[초기식;]
do {          무조건 일단 실행
              (조건 필요 없이 한번은 일단 실행)
    수행될 문장;
    [증감식 or 분기문;]
} while(조건식);
```

do 안의 내용 먼저 실행

조건식 확인 후 true면 문장 수행,

false면 종료

while 뒤에 ; 꼭 필요

* while과 do~while의 차이점 :

do~while은 조건문이 true가 아니더라도 무조건 한 번 이상 수행

✓ do ~ while문 예시

```
int i = 1;        한 번은 일단 실행해봐야
                  조건을 비교할 수 있는 경우
do {
    System.out.println(i + "출력");
    i++;
} while(i <= 10);
```

✓ 실행 결과

```
1 출력
2 출력
…
9 출력
10 출력
```

* { } 안에 조건을 벗어나게 할 연산(증감식, 분기문) 필요

▶ for문

✓ for

언제까지 반복?

안쪽에서 변수 변화시킴

```
for(초기식; 조건식; 증감식) {
        수행될 문장;
}
```

초기식 한번만 읽고
반복되는 동안에는 안읽는다.

1회전: 초기식 확인 후 조건식 확인
조건식이 true면 문장 수행
조건식이 false면 수행하지 않음

2회전: 증감식 연산 후 조건식 확인
조건식이 true면 문장 수행
조건식이 false면 수행하지 않음

* 2회전 이상부터는 모두 2회전과 동일하고
조건식이 false가 나올 때까지 문장 수행

✓ for문 예시

```
for(int i = 1; i <= 10; i++) {
        System.out.println(i + " 출력");
}
```

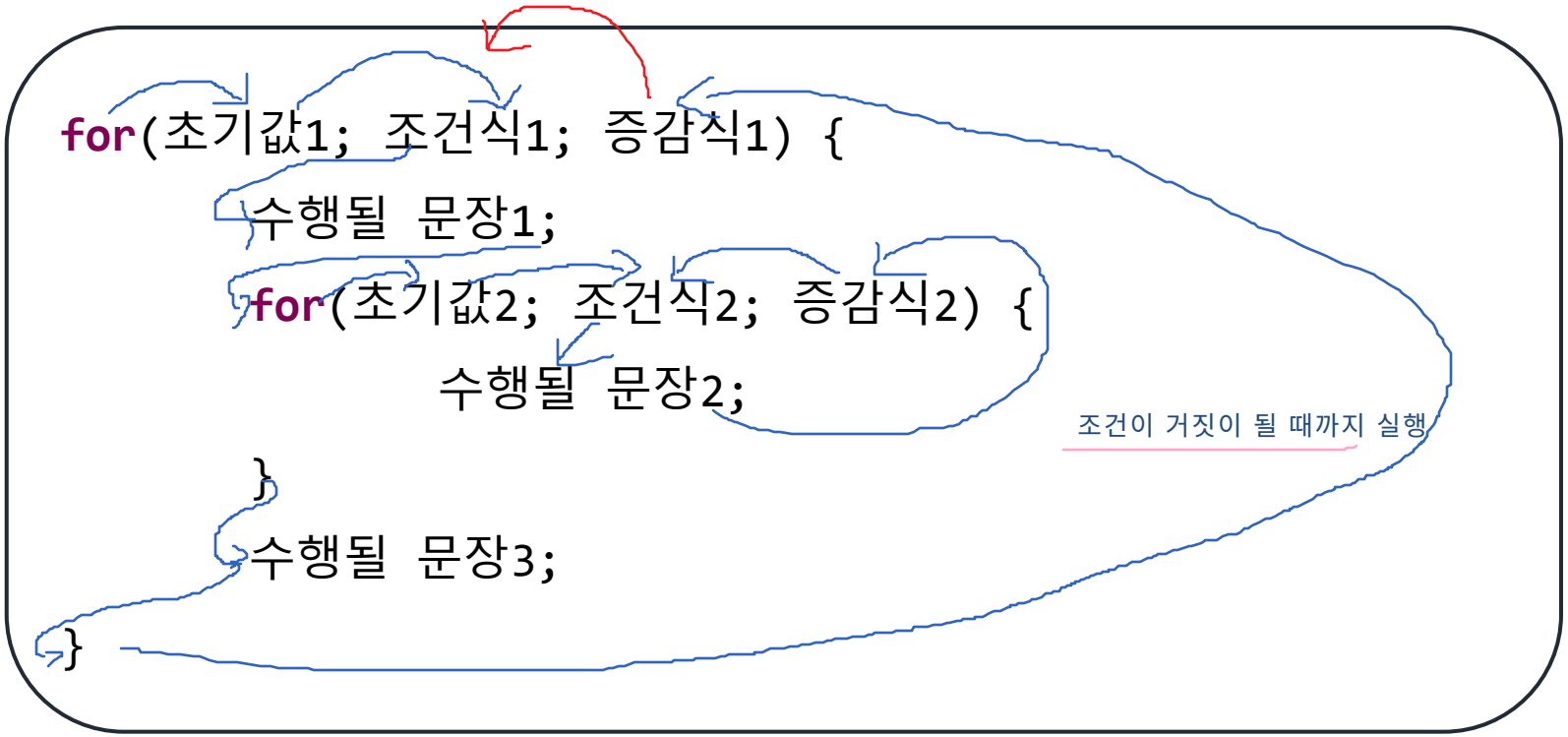
✓ 실행 결과

```
1 출력
2 출력
…
9 출력
10 출력
```

▶ 중첩 반복문

KH정보교육원

✓ 표현식



for문에 진입하면 수행될 문장1을 먼저 수행하고 두 번째 for문에 진입하면
조건식2가 false가 될 때까지 수행될 문장2를 수행 후 나오면
수행될 문장3을 수행하고 조건식1로 돌아와 true면 다시 반복

▶ 중첩 반복문

KH정보교육원

✓ 표현식

```
for(초기값1; 조건식1; 증감식1) {
        수행될 문장1;
        for(초기값2; 조건식2; 증감식2) {
                수행될 문장2;
                break;
        }
        수행될 문장3;
        [break;]
}
```

두 번째 for문에 break를 만날 경우 반복문을 나가 수행될 문장3을 수행 후 다시 첫 번째 for문을 실행하지만

마지막 break가 있다면 수행될 문장3을 수행 후 for문을 완전히 빠져나감

Chap03.
분기문

▶ 분기문

✓ break

반복문에서는 break문 자신이 포함된 가장 가까운 반복문을
빠져나가는 구문

✓ break문 예시

```
for(int i = 1;; i++) {
        System.out.println(i + " 출력");

        if(i >= 10) {
                break;
        }
}
```

▶ 분기문

✓ continue

반복문 내에서만 사용 가능하며 반복문 실행 시 continue 아래 부분은 실행하지 않고 반복문 다시 실행
for문의 경우 증감식으로 이동,
while(do~while)문의 경우 조건식으로 이동
전체 반복 중에 특정 조건을 만족하는 경우를 제외하고자 할 때 유용

✓ continue문 예시

```
for(int i = 1; i <= 10; i++) {
        if(i % 2 == 0) {
                continue;
        }
        System.out.println(i + " 출력");
}
```