



Today's lecture



`System.out.print('제어문');`

목차

Chap01. 조건문

Chap02. 반복문

Chap03. 분기문

Chap01. 조건문

조건문

프로그램 수행 흐름을 바꾸는 역할을 하는 제어문 중 하나로 조건에 따라 다른 문장이 수행되도록 함



조건문

프로그램 수행 흐름을 바꾸는 역할을 하는 제어문 중 하나로 조건에 따라 다른 문장이 수행되도록 함

✓ 조건문의 종류

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

✓ if

```
if(조건식) {  
    @  
}
```

조건식의 결과 값이 true면 @ 안의 내용 실행
false면 실행하지 않음

✓ if문 예시

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

if문

✓ if~else

```
if(조건식) {  
    ㉠  
} else {  
    ㉡  
}
```

조건식의 결과 값이 true면 ㉠ 안의 내용 실행
false면 ㉡ 안의 내용 실행

✓ if~else문 예시

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

if문

✓ if~else if~else

```
if(조건식1) {
    (a)
} else if(조건식2){
    (b)
} else {
    (c)
}
```

조건식1의 결과 값이 true면

① 안의 내용 실행

조건식2의 결과 값이 true면

② 안의 내용 실행

모두 false면

③ 안의 내용 실행

✓ 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.";
}
```

* if는 true, false와 상관 없이 조건절 실행,

if~else if~else는 조건문이 true면 이후 조건은 실행하지 않음

if문

✓ 중첩 if

```
if (조건식1) {  
    if (조건식2) {  
        if (조건식3) {  
            수행될 문장;  
        } else if (조건식4) {  
            수행될 문장;  
        } else {  
            수행될 문장;  
        }  
    } else {  
        수행될 문장;  
    }  
} else if (조건식5) {  
    수행될 문장;  
} else {  
    수행될 문장;  
}
```

✓ 중첩 if문 예시

```
if (month == 1 || month == 2 || month == 12) {  
    season = "겨울";  
    if (temperature <= -15) {  
        season += " 한파 경고";  
    } 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 경고";  
    } else if (temperature >= 33) {  
        season += " 폭염 주의보";  
    }  
} else if (month >= 9 && month <= 11) {  
    season = "가을";  
} else {  
    season = "해당하는 계절이 없습니다.";  
}
```

switch문

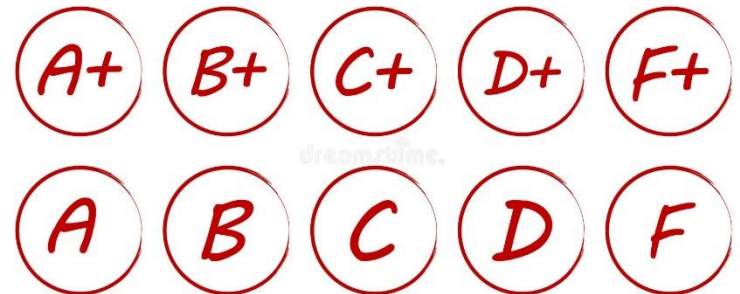
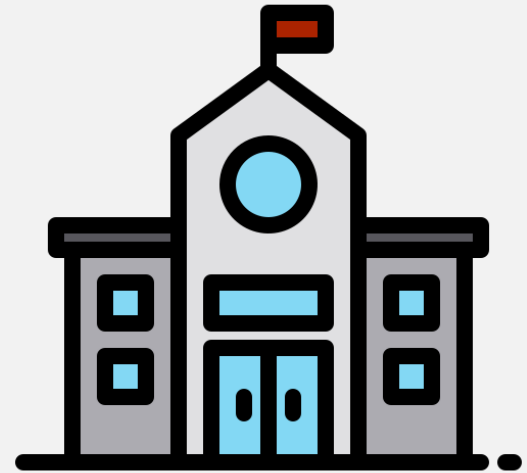
조건식 하나로 많은 경우의 수를
처리할 때 사용하며 이때 조건식의
결과는 정수 또는 문자, 문자열

조건식의 결과 값과 일치하는

case문으로 이동

default문은 일치하는

case문이 없을 때 수행(= else)



switch문

조건식 하나로 많은 경우의 수를
처리할 때 사용하며 이때 조건식의
결과는 정수 또는 문자, 문자열

조건식의 결과 값과 일치하는

case문으로 이동

default문은 일치하는

case문이 없을 때 수행(= else)

✓ 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";  
}
```

Chap02. 반복문

반복문

프로그램 수행 흐름을 바꾸는 역할을 하는 제어문 중 하나로 특정 문장들을 반복해서 수행하도록 함



반복문

프로그램 수행 흐름을 바꾸는 역할을 하는 제어문 중 하나로 특정 문장들을 반복해서 수행하도록 함

✓ 반복문의 종류

for문

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

while문

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

for문

✓ for

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

- 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 동일**하고
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

✓ 표현식

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

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

✓ 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

while문

✓ while

```
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

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

✓ 실행 결과

1 출력
2 출력
...
9 출력
10 출력

while문

✓ do ~ while

```
do {  
    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

* { } 안에 조건을 벗어나게 할 연산(증감식, 분기문) 필요

✓ do ~ while문 예시

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

✓ 실행 결과

```
1 출력  
2 출력  
...  
9 출력  
10 출력
```

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");  
}
```