

23.03.07 - 2강

3장 조건분기와 반복

프로그램의 흐름

코드를 실행시키는 순서를 **제어구조** 라고 한다.

대표적인 것으로는 **“순차”, “분기”, “반복”** 의 3가지가 있다

2장까지에 사용했던 “위에서 부터 순차적으로 한줄씩” 실행 것이 “순차” 이다

아무리 복잡한 프로그램이라도, 순차, 분기, 반복의 3가지 제어구조를 조합하면 만들 수 있다

블록

복수의 문장을 하나로 묶는 역할을 하기 위한 것

블록 작성 요령

룰 1 : 중괄호의 생략내용이 1행 밖에 없을 때는 중괄호를 생략해도 됨 - 안하는 것이 좋다

- 1) 가독성이 떨어진다.
- 2) 유지보수가 좋지 않다.

룰 2 : 블록 내에 선언한 변수의 수명블록 안에서 선언한 변수는 그 블록이 종료와 동시에 소멸한다

연산자

관계연산자

| 연산자 | 의미 |
|--------------|-----------------|
| == | 좌변과 우변이 같다 |
| != | 좌변과 우변이 다르다 |
| > | 좌변이 우변보다 크다 |
| < | 좌변이 우변보다 작다 |
| >= | 좌변이 우변보다 크거나 같다 |
| <= | 좌변이 우변보다 작거나 같다 |

연산 결과는 true / false의 boolean 값이 된다.

★문자열의 비교

Java에서 String 형의 비교에는 **특수한 작성 방법** 이 있다.

Java 에서는 문자열의 비교는 “==”를 사용하면 안 됨.

```
if (s=="str"){...} // 안됨
```

문자열의 비교를 할 때에는 반드시 다음과 같은 방법을 사용if (

```
if (s.equals("str")){...}
```

논리연산자

| 연산자 | 의미 |
|-----|-----------------------|
| && | 그리고 (모두 참이면 true) |
| | 또는 (둘 중 하나가 참이면 true) |

부정연산자

만약 ~이 아니면 을 조건식에 표현하고 싶으면 식의 앞에 **부정연산자 !** 를 붙인다

```
if (!(age == 10 )) {...}
```

수학의 표현과 Java 조건식의 표현(삼중비교)

x 는 10보다 크고, 20보다 작다

수학 => $10 < x < 20$

Java => $10 < x \ \&\& \ x < 20$

조건문

- if 명령어를 사용하면 “분기” 를 할 수 있다

- if 의 뒤에는 () 안에 “분기조건”을 쓴다
- 변수 clear 가 true 인지 아닌지를 체크를 할 때에는 “==” 을 사용
- 조건이 성립한다면, 해당 블록의 안쪽 코드가 실행됨
- 조건이 성립하지 않는다면, else 문의 블록 안쪽 코드가 실행됨

if문

원하는 조건을 if를 사용해서 나타냄.

추가적인 조건을 원하면 else if를 사용할 수 있다.

그 외의 경우는 else를 이용함.

예외(else if)가 많아지면 깔끔하지 않다

```
if (조건1){
    ...
} else if (조건2){
    ...
} else{
    ...
}
```

switch문

else if가 많은 조건문을 조금 더 깔끔하게 만들 수 있다.

break;를 조심할 것. - 단 의도적으로 성질을 이용하기 위해 사용하지 않을 수 있다.

```
switch (조건변수){
    case 변수값1 {
        ...
    }
    case 변수값2 {
        ...
    }
    case 변수값3 {
        ...
    }
    default {
        ...
    }
}
```

반복문

- while 을 사용하면 “반복” 제어를 행할 수 있다
- while 의 뒤에는 () 안에 “반복 조건”을 쓴다
- 반복 조건이 성립하는 한 계속 블록 안의 코드가 반복 실행됨

while문

조건을 판단 후 반복

```
while (반복조건) {
    ...
}
```

dowhile문

1회 실행 후 조건을 보고 반복

```
do {
    ...
} while(반복조건);
```

for문

```
for (반복변수 초기화; 조건식; 반복처리){
    ...
}
```

1. 루프 변수의 이름은 자유
2. 변수는 블록내에서 이용 가능
3. 블록 밖에서는 이용 불가
4. 초기값, 조건, 반복처리는 자유롭게 설정 가능하다.

```
for (int i = 1; i < 10; i++) { ... }
```

```
for (int i = 0; i < 10; i += 2) { ... }
```

```
for (int i = 10; i > 0; i--) { ... }
```

```
for (; i < 10; i++) { ... }
```

```
for (int i = 0; i < 10; ) { ... }
```

반복문의 제어

break 문 (중단)

for 문을 종료

continue 문 (건너뛰기)

이번만 중단하고, 계속 해서 루프를 진행

무한루프

무한하게 반복하는 제어구조

- while (true) { ... }
- for (;;) { ... }

탈출 조건을 잘 설정하는 것이 중요하다.

연습문제 3-1

다음 문장을 Java 조건식으로 기술하시오

1. 변수 `weight` 의 값이 60과 같다
2. 변수 `age1` 과 `age2` 의 합계를 2배 한 것이 60을 넘는다
3. 변수 `age` 가 홀수다
4. 변수 `name` 에 저장된 문자열이 "스마트"와 같다

```
if (weight == 60) {  
}  
if ((age1 + age2) * 2 > 60) {  
}  
if (age % 2 != 0) {  
}  
if (name.equals("스마트")) {  
}  
boolean isOdd = age % 2 == 1;  
boolean isEven = !isOdd;
```

연습문제 3-2

다음 중 조건식(if문의)으로 사용 할 수 있는 것을 고르시오

1. `cost = 300 * 1.05`
2. `3`
3. `age != 30`
4. `true`
5. `b + 5 < 20`
6. `gender = true`

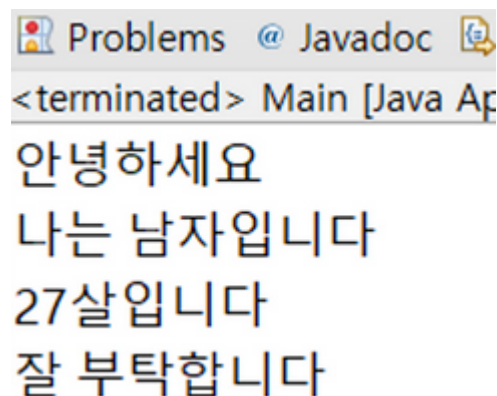
1. =은 대입 연산자이므로 조건문이 될 수 없다.
2. 비교연산이 없으므로 조건문이 될 수 없다. (boolean은 따로 존재한다.)
3. 가능
4. 가능
5. 가능
6. 실행은 가능하다. (gender에 true가 대입되어 true 조건이 된다. 단, 옳지 않은 코드이다.)

Java 프로그램으로 작성하시오

1. int형 변수 `gender` 를 선언하고, 0 또는 1을 대입한다 (어떤 것이라도 상관없음)
또한, int 형 변수 `age` 를 선언하고, 적당한 숫자를 대입한다.
2. 화면에 "안녕하세요" 를 표시한다
3. 만약 변수 `gender` 가 0이면 "나는 남자입니다", 그렇지 않으면 "나는 여자입니다" 를 표시한다
4. 만약 변수 `gender` 가 남자이면 `age` 변수의 값을 표시하고, 뒤에 "살입니다" 를 붙여서 표시한다.
5. 마지막으로 "잘 부탁드립니다" 를 표시한다

```
package com.example;

public class Main {
    public static void main(String[] args) {
        int gender = 0;
        int age = 27;
        System.out.println("안녕하세요");
        if (gender == 0) {
            System.out.println("나는 남자입니다");
            System.out.println(age + "살입니다");
        } else {
            System.out.println("나는 여자입니다");
        }
        System.out.println("잘 부탁드립니다");
    }
}
```




```

1 public class Main {
2     public static void main(String[] args) {
3         boolean clear = true;
4         if (clear == true) {
5             System.out.println("세탁을 한다");
6             System.out.println("산책을 한다");
7         } else
8             System.out.println("영화를 본다");
9     }
10 }

```

```

7         } else
8             System.out.println("영화를 본다");
9             System.out.println("잔다");
10    }
11 }

```

3행의 `clear` 가 `false` 일 경우 "영화를 본다" 다음에 "잔다" 를 표시하기 위해, 9행에 "잔다"를 표시하는 행을 추가하였다.

하지만 프로그램이 의도한 대로 움직이지 않았다. 어느 부분의 오류 때문에 어떤 현상이 발생하는지 생각해 보시오.
그리고 프로그램을 수정 하시오.

```

package com.example;

public class Main {
    public static void main(String[] args) {
        boolean clear = true;
        if (clear == true) {
            System.out.println("세탁을 한다");
            System.out.println("산책을 한다");
        } else
            System.out.println("영화를 본다");
        System.out.println("잔다");
    }
}

```

Problems @ Javadoc
<terminated> Main [Java]
세탁을 한다
산책을 한다
잔다

Problems @ Javadoc
<terminated> Main [Java]
영화를 본다
잔다

true

false

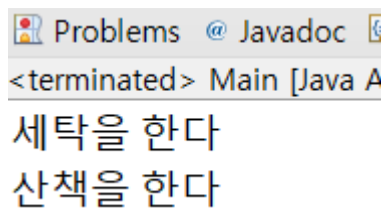
9행의 `System.out.println("잔다");`가 `else`문 안에 있지 않다.

따라서 `clear`의 값과 무관하게 항상 "잔다"가 출력된다.

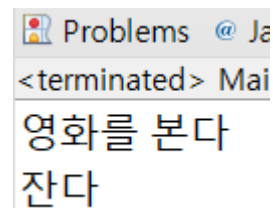
이를 수정하기 위해 중괄호({})를 추가한다.

```
package com.example;

public class Main {
    public static void main(String[] args) {
        boolean clear = true;
        if (clear == true) {
            System.out.println("세탁을 한다");
            System.out.println("산책을 한다");
        } else {
            System.out.println("영화를 본다");
            System.out.println("잔다");
        }
    }
}
```



Problems @ Javadoc
<terminated> Main [Java A]
세탁을 한다
산책을 한다



Problems @ Javadoc
<terminated> Main [Java A]
영화를 본다
잔다

true

false

연습문제 3-5

switch 문을 이용하여 다음 조건을 만족하는 프로그램을 작성하시오

1. 화면에 “[메뉴] 1:검색 2:등록 3:삭제 4:변경 >” 을 표시한다
2. 키보드로 숫자를 입력하고, 변수 **selected** 에 대입한다.
3. 만약 변수 **selected** 가 1 이면 “검색합니다”, 2이면 “등록합니다”, 3이면 “삭제합니다”, 4이면 “변경합니다”를 표시한다
4. **selected** 가 1 부터 4 사이의 값이 아니라면 아무것도 하지 않는다

```
package com.example;

import java.util.*;

public class Main {
    public static void main(String[] args) {
```

```

Scanner sc = new Scanner(System.in);
System.out.println("[메뉴] 1:검색 2:등록 3:삭제 4:변경 >");
int selected = sc.nextInt();
switch (selected) {
    case 1:
        System.out.println("검색합니다");
        break;
    case 2:
        System.out.println("등록합니다");
        break;
    case 3:
        System.out.println("삭제합니다");
        break;
    case 4:
        System.out.println("변경합니다");
        break;
    default: {
    }
}
}
}

```

Problems @ Javadoc Declaration Console ×
 <terminated> Main [Java Application] C:\Users\wadmin\>
 [메뉴] 1:검색 2:등록 3:삭제 4:변경 >
 1
 검색합니다

Problems @ Javadoc Declaration Console ×
 <terminated> Main [Java Application] C:\Users\wadmin\>
 [메뉴] 1:검색 2:등록 3:삭제 4:변경 >
 2
 등록합니다

Problems @ Javadoc Declaration Console ×
 <terminated> Main [Java Application] C:\Users\wadmin\>
 [메뉴] 1:검색 2:등록 3:삭제 4:변경 >
 3
 삭제합니다

Problems @ Javadoc Declaration Console ×
 <terminated> Main [Java Application] C:\Users\wadmin\>
 [메뉴] 1:검색 2:등록 3:삭제 4:변경 >
 4
 변경합니다

Problems @ Javadoc Declaration Console ×
 <terminated> Main [Java Application] C:\Users\wadmin\>
 [메뉴] 1:검색 2:등록 3:삭제 4:변경 >
 99

연습문제 3-6

다음 조건을 만족하는 프로그램을 작성하시오

1. 화면에 “[숫자 맞추기 게임]” 을 표시한다
2. 0 부터 9 까지의 정수 중에서 랜덤하게 수를 1개 생성해서 변수 **ans** 에 대입한다
3. **for** 문을 이용해 “5회 반복 하는 루프”를 만든다. 아래의 4. ~ 7. 은 루프 안에 기술 한다
4. 화면에 “0 ~ 9 사이의 숫자를 입력 하세요” 를 표시한다
5. 숫자를 입력해, 변수 **num** 에 대입한다
6. 만약 변수 **num** 이 변수 **ans** 와 같으면 “정답!” 이라고 화면에 표시하고 반복을 종료
7. 만약 변수 **num** 이 변수 **ans** 와 같지 않으면 “다릅니다”를 표시한다
8. 반복 블록의 바깥에, “게임을 종료합니다” 라고 화면에 표시한다

```
package com.example;

import java.util.*;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int ans = new java.util.Random().nextInt(10);
        System.out.println("[숫자 맞추기 게임]");
        for (int i = 0; i < 5; i++) {
            System.out.println("0 ~ 9 사이의 숫자를 입력 하세요");
            int num = sc.nextInt();
            if (num == ans) {
                System.out.println("정답!");
                break;
            } else {
                System.out.println("다릅니다");
            }
        }
        System.out.println("게임을 종료합니다");
    }
}
```

```
Problems @ Javadoc Declaration Console X
<terminated> Main [Java Application] C:\Users\wadmin\
[숫자 맞추기 게임]
0 ~ 9 사이의 숫자를 입력 하세요
3
다릅니다
0 ~ 9 사이의 숫자를 입력 하세요
5
다릅니다
0 ~ 9 사이의 숫자를 입력 하세요
7
다릅니다
0 ~ 9 사이의 숫자를 입력 하세요
9
정답!
게임을 종료합니다
```

quiz

입력 받을 문자열 수와 문자열을 입력받고 출력 포맷에 맞게 출력하세요.

입력 포맷

다음 포맷으로 입력

```
n // 입력 받을 문자열 수
s_1 // 문자열
s_2
..
s_n
```

출력 포맷

```
Hello s_1,s_2,..s_n.
```

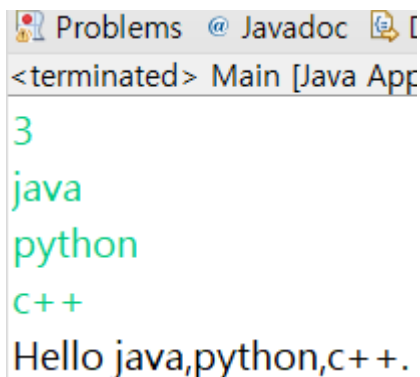
```

package com.example;

import java.util.*;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int num = sc.nextInt();
        String answer = "Hello ";
        for (int i = 0; i < num + 1; i++) {
            String tmp = sc.nextLine();
            if (i == 0) {
                continue;
            }
            if (i < num) {
                answer += tmp + ",";
            } else {
                answer += tmp + ".";
            }
        }
        System.out.println(answer);
    }
}

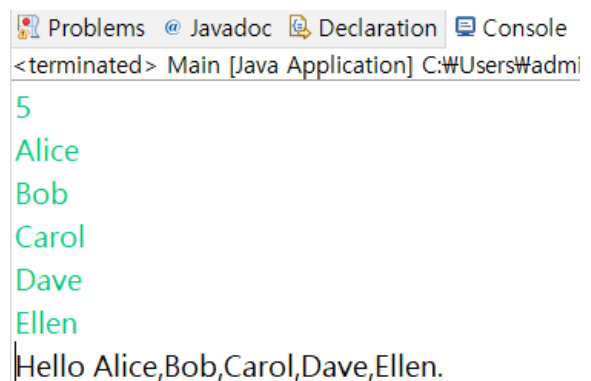
```



```

<terminated> Main [Java App
3
java
python
c++
Hello java,python,c++.

```



```

<terminated> Main [Java Application] C:\Users\Wadmi
5
Alice
Bob
Carol
Dave
Ellen
Hello Alice,Bob,Carol,Dave,Ellen.

```

Discussion

오류 발생

입력 예시를 하나씩 직접 입력하는 경우는 문제가 발생하지 않으나,

입력 예시를 통째로 복사하는 경우 오류가 발생하는 현상을 확인.

이를 개선하고 이유를 찾아보려했음.

입력 예2

```
5
Alice
Bob
Carol
Dave
Ellen
```

오류 발생 유형 코드 1 - for 문 안에서 새로운 객체를 생성하는 경우

```
package com.example;

import java.util.*;

public class Main {
    public static void main(String[] args) {
        int num = new Scanner(System.in).nextInt();
        String answer = "Hello ";
        for (int i = 0; i < num; i++) {
            String tmp = new Scanner(System.in).nextLine();
            if (i < num - 1) {
                answer += tmp + ",";
            } else {
                answer += tmp + ".";
            }
        }
        System.out.println(answer);
    }
}
```

이 경우, 생성된 Scanner 객체는 num+1개 이다.

이는 ~~코구려 수박도~~ chat gpt에서도 알 수 있는 사실이다.



위 코드에서 num이 5일 경우, Scanner 객체는 6개가 생성됩니다.



첫 번째 Scanner 객체는 `new Scanner(System.in)`을 통해 생성됩니다. 이 객체는 `nextInt()` 메소드를 호출하여 정수 값을 입력받습니다.

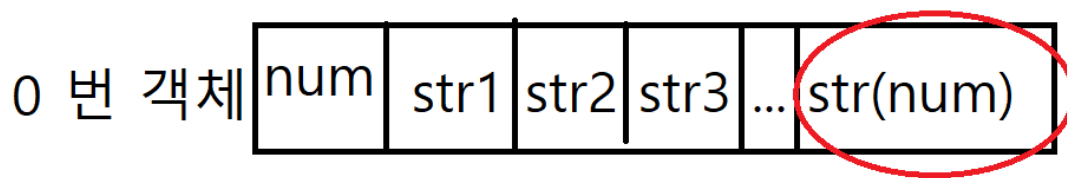
그 후, for 루프에서 Scanner 객체가 5번 생성됩니다. 각각의 객체는 `new Scanner(System.in)`을 통해 생성되며, `nextLine()` 메소드를 호출하여 문자열 값을 입력받습니다.

따라서 Scanner 객체는 6개가 생성되게 됩니다.

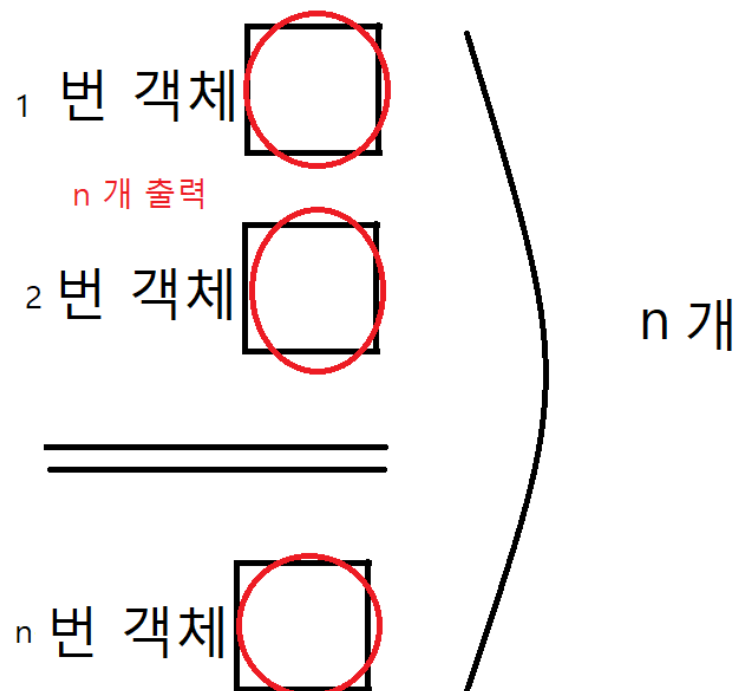
이때 입력 예시를 전부 복사하여 입력할 경우 다음과 같은 상황이 발생한다.

정확하게는 0번객체에 값이 배열처럼 누적되는 것이 아니라 덮어쓰워진다.

하지만 이해의 편의상 그림과 같이 표현한다. (이후로도 마찬가지로)



for 문 시작 (n개)




chat gpt에 의하면, 각 객체의 구조와 상태는 다음과 같다.

첫 번째 Scanner 객체를 통해 입력된 값인 "5WnaWnbWncWndWneWn"은 다섯 개의 문자열과 개행 문자로 이루어져 있습니다. 따라서 첫 번째 Scanner 객체는 개행 문자를 포함한 전체 문자열을 읽어들이고, 이후에 `nextInt()` 메소드를 호출하여 첫 번째 정수 값을 읽어들이는 것입니다. 이후에는 이 객체는 사용되지 않게 됩니다.

for 루프 내에서 생성되는 다섯 개의 Scanner 객체는 각각 `nextLine()` 메소드를 호출하여 입력된 문자열 값을 읽어들이는 것입니다. 이들 객체는 각각 n1, n2, n3, n4, n5 변수에 저장되며, 다음과 같은 구조를 갖습니다.

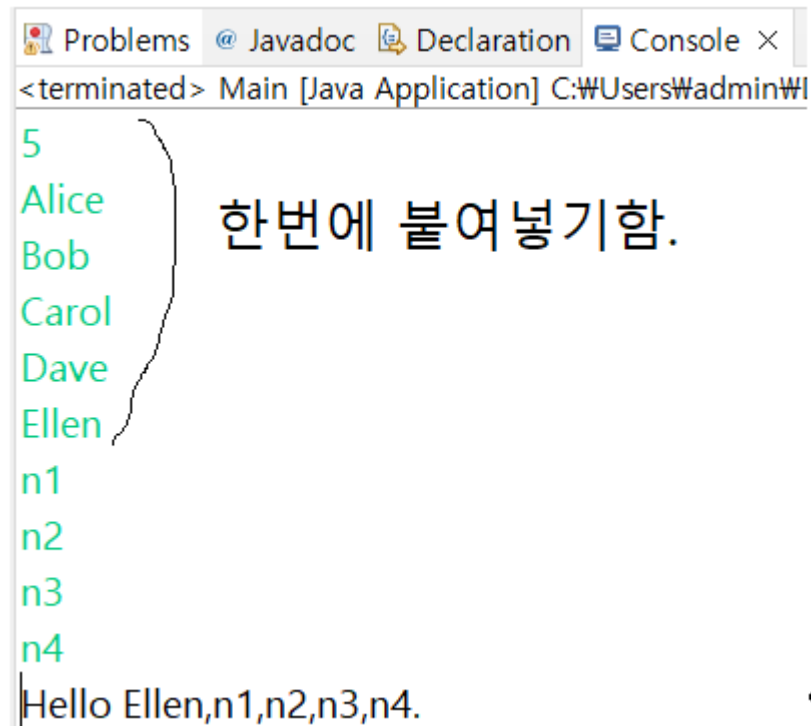
makefile

 Copy code

```
n1 = "a"
n2 = "b"
n3 = "c"
n4 = "d"
n5 = "e"
```

따라서, for 루프 내에서 생성되는 다섯 개의 Scanner 객체는 모두 `System.in`을 입력 소스로 사용하며, 각각의 객체는 `nextLine()` 메소드를 호출하여 입력된 문자열 값을

해결되지 않는 의문



```
<terminated> Main [Java Application] C:\Users\admin\Wl
5
Alice
Bob
Carol
Dave
Ellen
n1
n2
n3
n4
Hello Ellen,n1,n2,n3,n4.
```

그런데 출력 결과는 위와 같다.

예측되는 출력값은 n1,n2,n3,n4,n5여야 한다.

즉 맨 처음 생성된 객체가 한번 더 사용되고 6번째 객체가 생성되기 전에 반복문이 종료된다.

이에 대한 해답을 찾지 못해 매우 찝찝하다.

오류 발생 유형 코드 2 - 정수 입력을 위한 객체와 문자열 입력을 위한 객체를 분리한 경우

```
package com.example;

import java.util.*;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int num = new Scanner(System.in).nextInt();
        String answer = "Hello ";
        for (int i = 0; i < num; i++) {
            String tmp = sc.nextLine();
            if (i < num - 1) {
                answer += tmp + ",";
            } else {
                answer += tmp + ".";
            }
        }
    }
}
```

```

        System.out.println(answer);
    }
}

```

이 경우, 생성된 Scanner 객체는 2개 이다.

이 역시도 ~~코구려~~ 수박도 chat gpt에서 알 수 있는 사실이다.

위 코드에서는 두 개의 Scanner 객체가 생성됩니다.

첫 번째 Scanner 객체는 다음과 같이 생성됩니다.

```

java
Scanner sc = new Scanner(System.in);

```

이것은 표준 입력에서 데이터를 읽기 위한 Scanner 객체입니다.

두 번째 Scanner 객체는 다음과 같이 생성됩니다.

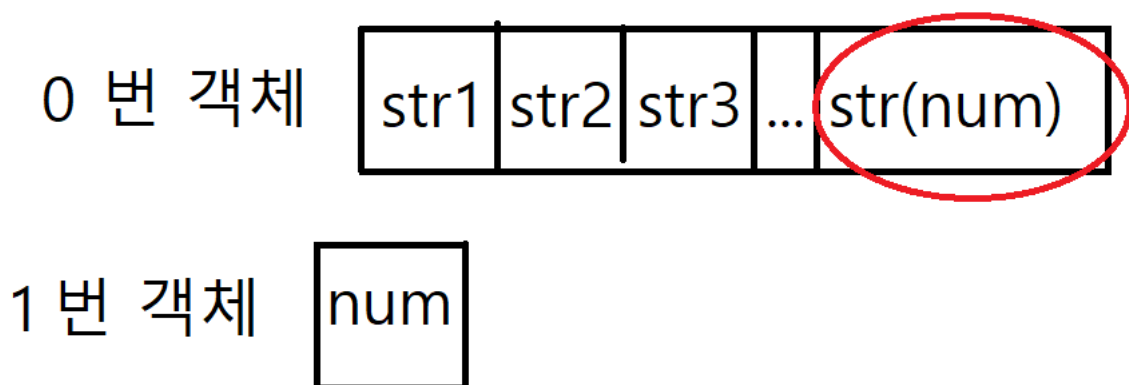
```

java
int num = new Scanner(System.in).nextInt();

```



이것은 표준 입력에서 int 값을 읽고 num 변수에 저장하기 위한 Scanner 객체입니다. 이 객체는 한 번 사용한 후에 더 이상 사용되지 않습니다.

이때 입력 예시를 전부 복사하여 입력할 경우 다음과 같은 상황이 발생한다.




chat gpt에 의하면, 각 객체의 구조와 상태는 다음과 같다.



입력이 "5, AliceWnBobWnCarolWnDaveWnEllen"인 경우, 첫 번째 Scanner 객체는 다음과 같   이 구성됩니다.

java


 Copy code

```
Scanner sc = new Scanner(System.in);
```

이 객체는 사용자가 입력한 값을 읽기 위한 Scanner 객체입니다. "System.in"은 InputStream이며, 이 객체는 InputStream에서 데이터를 읽어들이습니다.

두 번째 Scanner 객체는 다음과 같이 구성됩니다.

java


 Copy code

```
int num = new Scanner(System.in).nextInt();
```

이 코드는 사용자가 입력한 int 값을 읽기 위해 Scanner 객체를 생성합니다. 입력 값 "5"는 사용자가 입력한 첫 번째 값입니다. 이 객체는 System.in에서 데이터를 읽어들이고, nextInt() 메서드를 호출하여 사용자가 입력한 int 값을 읽어들이습니다.

반복문 내부의 Scanner 객체는 다음과 같이 구성됩니다.

java

 Copy code

```
String tmp = sc.nextLine();
```

이 코드는 사용자가 입력한 각 줄을 읽기 위한 Scanner 객체입니다. 첫 번째 루프에서는 "Alice"를 읽어들이고, 두 번째 루프에서는 "Bob"을 읽어들이습니다. 그런 다음, 이와 같은 방식으로 각 줄을 읽어들이습니다.

따라서 이 코드에서는 3개의 Scanner 객체가 생성됩니다. 첫 번째는 사용자가 입력한 값을 읽기 위한 객체이고, 두 번째는 사용자가 입력한 int 값을 읽기 위한 객체입니다. 반복문 내부에서는 Scanner 객체가 한 번 더 생성되어 각 줄을 읽어들이기 위한 객체입니다.

갑자기 객체가 3개라고 ~~옹졸하게~~ 말을 바꿨다

해결되지 않는 의문

```
Problems @ Javadoc Declaration Console X
<terminated> Main [Java Application] C:\Users\admin\W
5
Alice
Bob
Carol
Dave
Ellen
n1
n2
n3
n4
Hello Ellen,n1,n2,n3,n4.
```

한번에 붙여넣기함.

그런데 출력 결과는 위의 코드와 같다.

그런데 입력을 바꾸면 다른 결과가 나온다.

```
Problems @ Javadoc Declaration Console
<terminated> Main [Java Application] C:\Users\wadr
5
Alice
Bob
Carol
Dave
Ellen
Hello Alice,Bob,Carol,Dave,Ellen.
```

한번에 입력함

이 경우, 예상과 같이 2개의 객체가 생성되어 정상적인 출력형태가 나온다.

해결방법

객체를 한번만 생성하여 이를 사용한다.

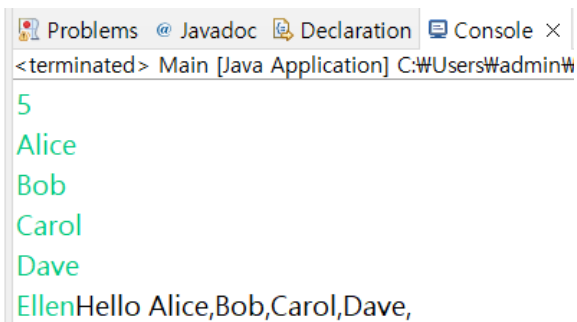
이때, Scanner 객체의 길이는 6이고, 0번째 값은 num이므로, 반복문을 num + 1번 반복하고, 첫회는 생략해야한다.

```

for (int i = 0; i < num + 1; i++) {
    String tmp = sc.nextLine();
    if (i == 0) {
        continue;
    }
    if (i < num) {
        answer += tmp + ",";
    } else {
        answer += tmp + ".";
    }
}

```

만약 반복문을 num회 반복할 경우 다음과 같은 결과가 나온다.



```

5
Alice
Bob
Carol
Dave
EllenHello Alice,Bob,Carol,Dave,

```

i의 값이 1,2,3,4일때 4가지의 값이 출력된다.

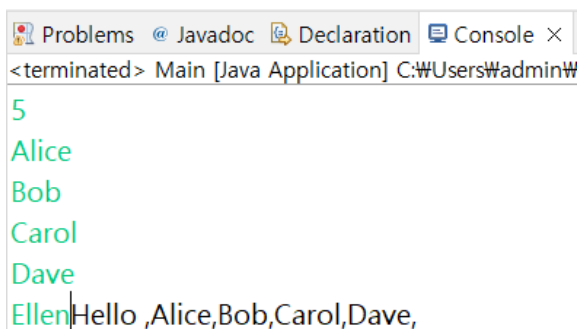
입력을 위해 enter를 누르기도 전에 개행문자의 갯수가 충족되어 반복문이 끝난다.

만약 첫회 생략을 지운 경우 다음과 같은 결과가 나온다.

```

for (int i = 0; i < num + 1; i++) {
    String tmp = sc.nextLine();
    if (i < num) {
        answer += tmp + ",";
    } else {
        answer += tmp + ".";
    }
}

```



```

5
Alice
Bob
Carol
Dave
EllenHello ,Alice,Bob,Carol,Dave,

```

i의 값이 0,1,2,3,4일때 5가지의 값이 출력된다.

입력을 위해 enter를 누르기도 전에 개행문자의 갯수가 충족되어 반복문이 끝난다.

0번째 값은 nextInt()에 의해 5\n이 분리되어 5가 num에 저장되고 \n만 남았으므로 공백이 출력된다.

객체를 한번 생성할 시 가장 범하기 쉬운 실수이다.

해결되지 않은 의문에 대한 뇌피셜

한 번에 값을 입력한 경우, `Scanner.nextInt()`을 통해 5만 먼저 `num`에 따로 저장된 후, 그 이후의 문자열 `Alice\nBob\nCarol\nDave\nEllen`이 순차적으로 덮어쓰워진다.

- 이 예제에서, 사용자가 "5_(Spacebar)"를 입력했다고 하자.
- `nextInt()` 메서드는 입력값을 읽을 때 공백문자(여기서는 **Spacebar**)를 제외해서 읽으므로, "5"만 읽어들이는다.
- 그렇게 되면 입력 버퍼에는 "5"가 나가고 "_"만 남게 된다.
- `a`의 출력 결과를 보면, 애초에 `nextInt()`가 입력값을 읽을 때 5만 읽어서 반환하여 `a`에 저장했으므로 출력값도 "5_"가 아닌 "5"가 된다.

즉, `for`문의 첫번째 회차에서 새로 생성된 `Scanner` 객체가 5 이후의 값들이 덮어쓰워져 최종적으로 `Ellen`일 것이다. 이후 2회차에서 `n1`, 3회차에서 `n2`,... 이런식으로 저장되면 출력 결과가 위와 같이 나오는 것이 설명 가능하다.

- 이 예제에서 사용자가 `num`에는 "5"를, `str`에는 "I love you"를 저장하길 원한다고 하자.
- 사용자가 `nextInt()` 메서드에 "5Wn(Enter)"를 입력했다고 하자.
- `nextInt()` 메서드는 입력값을 읽을 때 공백문자(여기서는 **Enter**)를 제외해서 읽으므로, "5"만 읽어들이는다.
- 그렇게 되면 입력 버퍼에는 "5"가 나가고 "Wn"만 남게 된다.
- `nextLine()` 메서드는 입력값을 읽을 때 (입력 버퍼에 저장되어있는) 공백문자를 포함하여 읽는다.
즉, `nextInt()`로 인해 입력 버퍼에 저장된 "Wn"까지 읽어들이는 것이다.
- 그런데 `nextLine()`은 **Enter** 단위로 값을 읽으므로, 입력 버퍼에 저장된 "Wn"를 읽은 후 입력값을 모두 읽어들이었다고 생각하고 넘어가버린다.
즉 "I love you"를 입력하기도 전에 입력을 끝내버리는 것이다.
- `nextLine()`은 반환시에는 "Wn"를 버리고 반환하므로, 결국에는 아무것도 `return` 되지 않게 된다.

결론

5\nAlice\nBob\nCarol\nDave\nEllen을 입력한 경우 다음과 같은 과정으로 처리된다.

0) System.in을 통해 5\nAlice\nBob\nCarol\nDave\nEllen을 입력받아 저장한다.

1) nextInt()가 첫번째 \n앞 숫자 5를 읽어 num에 저장한다. 뒤의 입력은 아직 처리되지 않았다.

num = 5

Scanner가 처리하지 않은 입력 : \nAlice\nBob\nCarol\nDave\nEllen

이때 각 코드에 따라 결과가 달라진다.

제출 코드 :

for 문에 돌입하여 nextLine()을 수행한다. 다음과 같은 결과가 나온다.

공백(개행문자 앞에 아무것도 없어서),Alice,Bob,Carol,Dave,Ellen

공백을 무시하므로 문제 없이 정상 출력된다.

오류 1번 코드 : 가장 마지막의 Ellen이 1번 객체에 저장되고 이후 객체를 매번 새로 만들어 입력을 받는다.

오류 2번 코드 : 가장 마지막의 Ellen이 0번 객체에서(반복횟수를 위한 객체보다 먼저 선언됨) 처리되고 이후 해당 객체에서 입력을 받는다. 따라서 객체의 갯수는 이론과 같이 2개로 유지되나, 입력을 4번 더 받아야 해 1번과 같은 결과가 나온 것이다.

또 다른 해답

따라서 nextInt()이후에 개행문자를 무시할 수 있게 nextLine()을 한번 실행해줘도 같은 결과를 얻을 수 있다.

```
package com.example;

import java.util.*;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int num = sc.nextInt();
        sc.nextLine();
        String answer = "Hello ";
        for (int i = 0; i < num; i++) {
            String tmp = sc.nextLine();
            if (i < num - 1) {
                answer += tmp + ",";
            }
        }
    }
}
```



```
        } else {  
            answer += tmp + ".";  
        }  
    }  
    System.out.println(answer);  
}  
}
```

참고 : https://velog.io/@cse_pebb/Java-Scanner-메서드-총정리