2023-03-08

final 키워드

final이 변수나 메서드나 클래스 앞에 붙으면, 해당 엔티티(변수/메서드/클래스)는 변경할 수 없는 엔티티가 됨. 변수명은 주로 대문자로 작성.

```
final char STAR = '*';
```

```
// 난수 범위 설정
(int) Math.random() * (최댓값-최소값+1) + 최소값
```

중첩 for문

for문 안에 for문을 넣어서 사용하는 반복문.

```
public static void main(String[] args) {
    final char STAR = '*';
    for (int i = 1; i <= 5; i++) {
      for (int j = 1; j <= 10; j++) {
        System.out.print(STAR);
      }
      System.out.println();
    }
  }
/* -- console 출력 --*/
**********
**********
**********
**********
**********
```

JAVA 5부터 추가된 for문

```
for(변수선언 : 배열 또는 컬렉션 개체)
```

배열이나 컬렉션 개체가 가지고있는 데이터 값들에 대한 반복 수행문장

While 문

```
while(조건식)
      반복 수행문장

while(true) {
  if(조건식)
    break;
      반복 수행문장
}
  수행문장
```

For문 & While문

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

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

Break

```
public static void main(String[] args) {
    int result;
    // 라벨: 이름을 지정해주는 것
    unico: for(int dan = 1; dan <= 9; dan++) {
      for(int num=1; num <= 9; num++) {
        result = dan*num;
        if (result >= 30)
          break unico;
        // unico라는 라벨을 가진 반복문을 break한다.
        // 만일 라벨을 지정해주지 않고, break만 사용한다면 가장 가까운 하나만 나간다
        System.out.print(dan + "x" + num + "=" + result + "\t");
      }
      System.out.println();
    }
  }
```

라벨을 활용하면 같은 반복문 안에서 특정한 반복문의 조건을 만족할 때 break하도록 사용 가능하다.

라벨은 변수값과 같이 임의로 설정할 수 있으나 : 를 함께 사용해야한다.

break : 반복문 종료

Continue

```
while(true) {
      num = (int)(Math.random()*31);
      if (num == 0) {
        System.out.println("\n종료");
        break;
      }
      if (num > 26) {
        System.out.println('@');
        continue;
      }
      System.out.printf("%d(%c)", num, (char)(96+num));

    }
```

continue : 다시 반복문으로 돌아가기

System.out.printf

```
// 포맷문자(%로 시작하는 문자)
%d: 10진수
%x: 8진수
%o: 16진수
%c: char
0X%x : 16진수로 변환
%b: boolean
%f: float
%s: string

// 포맷문자의 개수와 순서에 맞게 아규먼트를 명시.
System.out.printf("4. %d %x %o %c\n", 100, 100, 100, 100);
// n$는 n번째 아규먼트를 명시함
System.out.printf("5. %d %1$x %1$o %1$c\n", 100);
```

```
int exprSum(int a, int b)

// 가변아규먼트 int...
int exprSum(int...p)

int exprSum(int[] p)
```

가변아규먼트를 사용하면 전달하고자 하는 것의 정수 개수에 상관없이 모두 전달할 수 있다.

System.out

표준출력장치(화면)로 출력하는 기능을 가지고 있는 API

System.in

표준입력장치(키보드)로 입력하는 기능을 가지고 있는 API