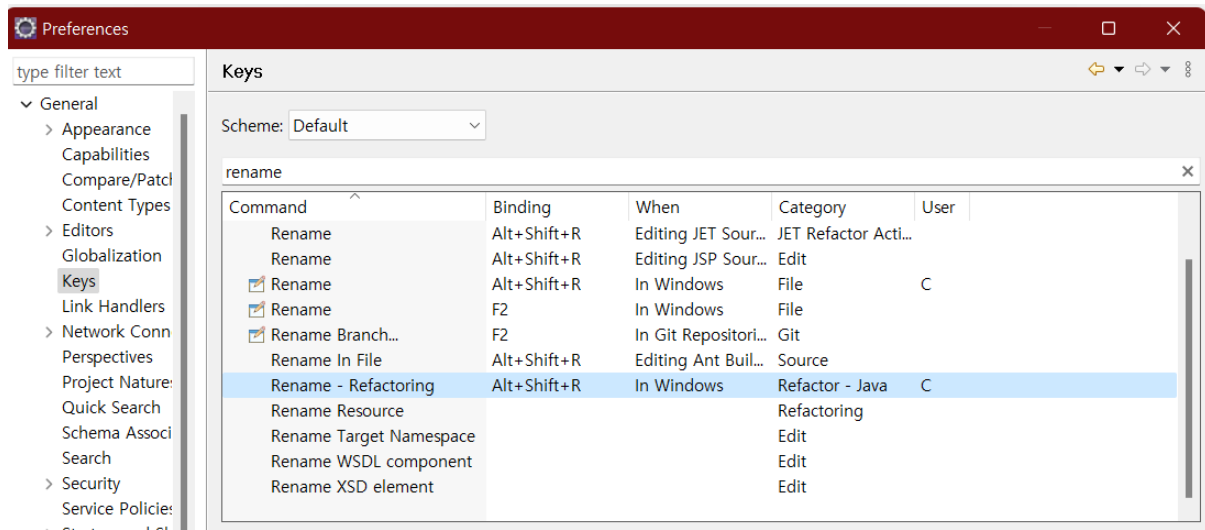


# 23.03.10 5강

## 어제 과제 피드백

변수 이름 ⇒ 편의상 입력을 answer라고 한것 바꾸면 편함

### 단축키 설정 방법



window - preferences - general - keys

## 복수 클래스를 사용한 개발

### 소스 파일의 분리

대규모 개발에서는 혼자서 개발이 어렵고 분담을 하여 소스를 **모듈화** 해야 한다.

1개의 소스파일로는 개발의 한계를 만날 것이다.

함수만 나누는 것이 아니라 클래스 자체를 쪼개서 만들기

### 패키지 (package)

각 클래스를 패키지 (package)에 소속시켜, 분류, 관리

main() 메소드의 라인수가 늘어나면 복수의 메소드로 분리 ->

메소드 수가 늘어나면 복수의 클래스로 분리 ->

클래스 수가 늘어나면 복수의 패키지로 분리

## import

다른 패키지의 클래스와 메소드를 호출하기 위해선 import해준다.

```
1 package calcapp.main;
2
3 public class Calc {
4     public static void main(String[] args) {
5         int a = 10;
6         int b = 2;
7         int total = calcapp.logics.CalcLogic.add(a, b);
8         int delta = calcapp.logics.CalcLogic.minus(a, b);
9         System.out.println("더하면 " + total + ", 빼면 " + delta);
10    }
11 }
```

```
1 package calcapp.main;
2
3 import calcapp.logics.CalcLogic;
4
5 public class Calc {
6     public static void main(String[] args) {
7         int a = 10;
8         int b = 2;
9         int total = CalcLogic.add(a, b);
10        int delta = CalcLogic.minus(a, b);
11        System.out.println("더하면 " + total + ", 빼면 " + delta);
12    }
13 }
```

훨씬 간결해진다

## 참조형의 비교 equals 하는 이유

참조형을 비교할 때는 ==이 아닌 .equals로 비교한다 ⇒ 주소이므로

갓이썬은 ==을 .equals로 재정의가 됨

## JAVA API(중요)

API (Application Programming Interface)

쓰기 편한 기능들을 미리 만들어 둔것. (라이브러리, 프레임워크와 유사)

## 라이브러리, 프레임워크, API 차이



라이브러리, 프레임워크 및 API는 모두 소프트웨어 개발에서 중요한 개념이지만, 서로 다른 역할을 합니다.

라이브러리는 소프트웨어 개발을 위해 사용되는 코드 모음입니다. 라이브러리는 개발자가 필요에 따라 호출할 수 있는 함수 및 기능을 제공합니다. 라이브러리는 개발자가 코드를 직접 작성하지 않고도 기능을 확장하고 개선하는 데 도움이 됩니다. 예를 들어, NumPy는 파이썬에서 사용되는 라이브러리로, 수학 및 과학 연산을 처리하는 데 사용됩니다.

프레임워크는 소프트웨어 개발을 위한 전체적인 구조를 제공하는 라이브러리의 집합입니다. 프레임워크는 개발자에게 작업 흐름을 정의하고 규정하며, 개발자는 이를 따라야 합니다. 프레임워크는 일반적으로 웹 개발, 애플리케이션 개발 등과 같은 큰 규모의 프로젝트에서 사용됩니다. 예를 들어, Django는 파이썬 웹 프레임워크로, 웹 애플리케이션 개발을 쉽게 만들어줍니다.

API는 응용 프로그램 간의 상호 작용을 위한 프로그래밍 인터페이스입니다. API는 응용 프로그램에서 데이터를 보내고 받는 데 사용됩니다. API는 일반적으로 웹 서비스에서 사용되며, 데이터의 송수신을 관리하고, 응용 프로그램 간의 상호 작용을 돕습니다. 예를 들어, Google Maps API는 지도와 위치 데이터를 제공하는 데 사용됩니다.

따라서, 간단히 말해, 라이브러리는 기능을 제공하고, 프레임워크는 구조를 제공하며, API는 인터페이스를 제공합니다.

## API 패키지의 종류

|           |                                 |
|-----------|---------------------------------|
| java.lang | Java 에서 가장 중요한 클래스군 (자동 import) |
| java.util | 프로그래밍을 편리하게 해 주는 유용한 클래스군       |
| java.math | 수학에 관한 클래스군                     |
| java.net  | 네트워크 통신등에 필요한 클래스군              |
| java.io   | 파일 입출력 등에 필요한 클래스군              |

## 출력 포맷

```

1 public class Main {
2     public static void main(String[] args) {
3         System.out.printf("%d 숫자를 나타냄\n", 1);
4         System.out.printf("%s 을 표시\n", "문자열");
5         System.out.printf("%3.2f 정수부분이 3자리, 소수점 아래가 2자리\n", 3.5f);
6         System.out.printf("%2d 2자리의 10진수 숫자 숫자이외의 부분은 공백\n", 1);
7         System.out.printf("%02d 2자리의 10진수 숫자 숫자이외의 부분은 0\n", 1);
8         System.out.printf("가\t나\t다 : %s의 예제\n", "탭");
9         System.out.printf("제 이름은 %s입니다. 나이는 %d살 키는 %3.1fcm 입니다", "오준석", 36, 173.3);
10    }
11 }

```

c언어의 printf 지원해준다.

## 문자열 조작

### 문자열처리

#### 문자열 합

“Hello” + “ Java”

=> “Hello Java”

```
String s1 = "Hello";
```

```
String s2 = " Java"
```

```
System.out.print(s1 + s2);|
```

#### substring

“HELLO”

=> “HE”

```
String s1 = "HELLO";  
  
System.out.println(s1.substring(0, 2));
```

replace

“HELLO”

=> “HEXXO”

```
String s1 = "HELLO";  
  
System.out.println(s1.replace("LL", "XX"));
```

split

“1,2,3”

=> “1”, “2”, “3”

```
String s1 = "1,2,3";  
  
String splited[] = s1.split(",");  
  
for (String s : splited) {  
    System.out.println(s);  
}
```

**toLowerCase**

“HELLO”

=> “hello”

```
String s1 = "HELLO";  
  
System.out.println(s1.toLowerCase());
```

## indexOf

“HELLO”

=> E 는 2번째 글자

```
String s1 = "HELLO";  
  
System.out.println(s1.indexOf('E'));
```

```
String s1 = "Java and JavaScript";  
  
System.out.println(s1.contains("Java")); // true  
System.out.println(s1.endsWith("Java")); // false  
System.out.println(s1.indexOf("Java")); // 0  
System.out.println(s1.lastIndexOf("Java")); // 9
```

contains() : 포함 관계  
endsWith() : 끝나는 단어가 맞는지  
indexOf() : 단어가 몇 번째에 있는지  
lastIndexOf() : 뒤에서 몇 번째에 단어가 있는지

## equals, equalsIgnoreCase

```
String s1 = "Java";
String s2 = "java";

if (s1.equals(s2)) {
    System.out.println("s1과 s2는 같다");
}

if (s1.equalsIgnoreCase(s2)) {
    System.out.println("s1과 s2는 대소문자 무시하면 같다");
}
```

## length, isEmpty

```
String s1 = "Java";

System.out.println(s1.length()); // 4

System.out.println(s1.isEmpty()); // false
```

length() : 길이  
isEmpty() : 길이가 0인지

## StringBuilder



append() 메서드로 결합한 결과를 내부 메모리(버퍼)에 담아 두고 toString()으로 결과를 얻음

```
StringBuilder sb = new StringBuilder();
for (int i = 0; i < 10000; i++) {
    sb.append("Java"); // 결합
}
String s = sb.toString(); // 완성된 결과
```

## 연습문제

### 연습문제 6-1

```
1 public class Main {
2     public static void main(String[] args) throws Exception {
3         System.out.println("3초간 기다림!");
4
5         // 3초간 기다림
6
7         System.out.println("끝");
8     }
9 }
```

java.lang.Thread 클래스를 조사하여, 프로그램을 3초간 멈추게 하는 프로그램을 완성하시오. throws Exception 에 대해서는 나중에 배우니 일단 무시하시오.

```
package com.example;

import java.lang.Thread;

public class Main {
    public static void main(String[] args) throws Exception {
        System.out.println("3초간 기다림!");

        // sleep은 ms 단위이므로 3초 대기하려면 3000 넣어야함
        Thread.sleep(3000);

        System.out.println("끝");
    }
}
```

```
Problems @ Javadoc Declaration Console X
Main [Java Application] C:\Users\wadmin\Desktop\wecli
3초간 기다림!
```

```
Problems @ Javadoc Declaration Console X
<terminated> Main [Java Application] C:\Users\wadmin
3초간 기다림!
끝
```

## 연습문제 6-2

구구단을 작성하시오

|            |            |            |            |            |            |            |            |
|------------|------------|------------|------------|------------|------------|------------|------------|
| 2 * 1 = 2  | 3 * 1 = 3  | 4 * 1 = 4  | 5 * 1 = 5  | 6 * 1 = 6  | 7 * 1 = 7  | 8 * 1 = 8  | 9 * 1 = 9  |
| 2 * 2 = 4  | 3 * 2 = 6  | 4 * 2 = 8  | 5 * 2 = 10 | 6 * 2 = 12 | 7 * 2 = 14 | 8 * 2 = 16 | 9 * 2 = 18 |
| 2 * 3 = 6  | 3 * 3 = 9  | 4 * 3 = 12 | 5 * 3 = 15 | 6 * 3 = 18 | 7 * 3 = 21 | 8 * 3 = 24 | 9 * 3 = 27 |
| 2 * 4 = 8  | 3 * 4 = 12 | 4 * 4 = 16 | 5 * 4 = 20 | 6 * 4 = 24 | 7 * 4 = 28 | 8 * 4 = 32 | 9 * 4 = 36 |
| 2 * 5 = 10 | 3 * 5 = 15 | 4 * 5 = 20 | 5 * 5 = 25 | 6 * 5 = 30 | 7 * 5 = 35 | 8 * 5 = 40 | 9 * 5 = 45 |
| 2 * 6 = 12 | 3 * 6 = 18 | 4 * 6 = 24 | 5 * 6 = 30 | 6 * 6 = 36 | 7 * 6 = 42 | 8 * 6 = 48 | 9 * 6 = 54 |
| 2 * 7 = 14 | 3 * 7 = 21 | 4 * 7 = 28 | 5 * 7 = 35 | 6 * 7 = 42 | 7 * 7 = 49 | 8 * 7 = 56 | 9 * 7 = 63 |
| 2 * 8 = 16 | 3 * 8 = 24 | 4 * 8 = 32 | 5 * 8 = 40 | 6 * 8 = 48 | 7 * 8 = 56 | 8 * 8 = 64 | 9 * 8 = 72 |
| 2 * 9 = 18 | 3 * 9 = 27 | 4 * 9 = 36 | 5 * 9 = 45 | 6 * 9 = 54 | 7 * 9 = 63 | 8 * 9 = 72 | 9 * 9 = 81 |

```
package com.example;

public class Main {
    public static void main(String[] args) throws Exception {
        String special = "\t";
        for (int i = 2; i < 10; i++) {
            for (int k = 1; k < 10; k++) {
                System.out.printf("%d * %d = %2d", k, i, k * i);
                if (k == 9) {
                    special = "\n";
                } else {
                    special = "\t";
                }
                System.out.printf("%s", special);
            }
        }
    }
}
```

```
Problems @ Javadoc Declaration Console X
<terminated> Main [Java Application] C:\Users\wadmin\Desktop\weclipse\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.5.v20221102-0933\jre\bin\javaw.exe (2023.3.10. 오전 11:34:45)
1 * 2 = 2    2 * 2 = 4    3 * 2 = 6    4 * 2 = 8    5 * 2 = 10   6 * 2 = 12   7 * 2 = 14   8 * 2 = 16   9 * 2 = 18
1 * 3 = 3    2 * 3 = 6    3 * 3 = 9    4 * 3 = 12   5 * 3 = 15   6 * 3 = 18   7 * 3 = 21   8 * 3 = 24   9 * 3 = 27
1 * 4 = 4    2 * 4 = 8    3 * 4 = 12   4 * 4 = 16   5 * 4 = 20   6 * 4 = 24   7 * 4 = 28   8 * 4 = 32   9 * 4 = 36
1 * 5 = 5    2 * 5 = 10   3 * 5 = 15   4 * 5 = 20   5 * 5 = 25   6 * 5 = 30   7 * 5 = 35   8 * 5 = 40   9 * 5 = 45
1 * 6 = 6    2 * 6 = 12   3 * 6 = 18   4 * 6 = 24   5 * 6 = 30   6 * 6 = 36   7 * 6 = 42   8 * 6 = 48   9 * 6 = 54
1 * 7 = 7    2 * 7 = 14   3 * 7 = 21   4 * 7 = 28   5 * 7 = 35   6 * 7 = 42   7 * 7 = 49   8 * 7 = 56   9 * 7 = 63
1 * 8 = 8    2 * 8 = 16   3 * 8 = 24   4 * 8 = 32   5 * 8 = 40   6 * 8 = 48   7 * 8 = 56   8 * 8 = 64   9 * 8 = 72
1 * 9 = 9    2 * 9 = 18   3 * 9 = 27   4 * 9 = 36   5 * 9 = 45   6 * 9 = 54   7 * 9 = 63   8 * 9 = 72   9 * 9 = 81
```

## 연습문제 6-3

전자시계 프로그램을 작성하시오



```
1:00 2:00 3:00 4:00 5:00 6:00 7:00 8:00 9:00 10:00 11:00 12:00
1:01 2:01 3:01 4:01 5:01 6:01 7:01 8:01 9:01 10:01 11:01 12:01
1:02 2:02 3:02 4:02 5:02 6:02 7:02 8:02 9:02 10:02 11:02 12:02
1:03 2:03 3:03 4:03 5:03 6:03 7:03 8:03 9:03 10:03 11:03 12:03
1:04 2:04 3:04 4:04 5:04 6:04 7:04 8:04 9:04 10:04 11:04 12:04
1:05 2:05 3:05 4:05 5:05 6:05 7:05 8:05 9:05 10:05 11:05 12:05
...
1:59 2:59 3:59 4:59 5:59 6:59 7:59 8:59 9:59 10:59 11:59 12:59
```

```
package com.example;

public class Main {
    public static void main(String[] args) throws Exception {
        String special = " ";
        for (int k = 0; k < 60; k++) {
            for (int i = 1; i < 13; i++) {
                System.out.printf("%2d:%02d", i, k);
                if (i == 12) {
                    special = "\n";
                } else {
                    special = " ";
                }
                System.out.printf("%s", special);
            }
        }
    }
}
```

Problems @ Javadoc Declaration Console ×

<terminated> Main [Java Application] C:\Users\wadmin\Desktop\weclipse\plugins\org.eclipse.justj.c

```
1:00 2:00 3:00 4:00 5:00 6:00 7:00 8:00 9:00 10:00 11:00 12:00
1:01 2:01 3:01 4:01 5:01 6:01 7:01 8:01 9:01 10:01 11:01 12:01
1:02 2:02 3:02 4:02 5:02 6:02 7:02 8:02 9:02 10:02 11:02 12:02
1:03 2:03 3:03 4:03 5:03 6:03 7:03 8:03 9:03 10:03 11:03 12:03
1:04 2:04 3:04 4:04 5:04 6:04 7:04 8:04 9:04 10:04 11:04 12:04
1:05 2:05 3:05 4:05 5:05 6:05 7:05 8:05 9:05 10:05 11:05 12:05
1:06 2:06 3:06 4:06 5:06 6:06 7:06 8:06 9:06 10:06 11:06 12:06
1:07 2:07 3:07 4:07 5:07 6:07 7:07 8:07 9:07 10:07 11:07 12:07
1:08 2:08 3:08 4:08 5:08 6:08 7:08 8:08 9:08 10:08 11:08 12:08
1:09 2:09 3:09 4:09 5:09 6:09 7:09 8:09 9:09 10:09 11:09 12:09
```

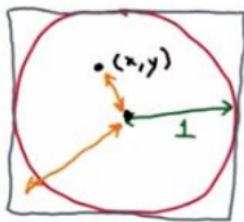
```

Problems @ Javadoc Declaration Console ×
<terminated> Main [Java Application] C:\Users\Wadmin\Desktop\weclipse\plugins\org.eclipse.justj.o
1:51 2:51 3:51 4:51 5:51 6:51 7:51 8:51 9:51 10:51 11:51 12:51
1:52 2:52 3:52 4:52 5:52 6:52 7:52 8:52 9:52 10:52 11:52 12:52
1:53 2:53 3:53 4:53 5:53 6:53 7:53 8:53 9:53 10:53 11:53 12:53
1:54 2:54 3:54 4:54 5:54 6:54 7:54 8:54 9:54 10:54 11:54 12:54
1:55 2:55 3:55 4:55 5:55 6:55 7:55 8:55 9:55 10:55 11:55 12:55
1:56 2:56 3:56 4:56 5:56 6:56 7:56 8:56 9:56 10:56 11:56 12:56
1:57 2:57 3:57 4:57 5:57 6:57 7:57 8:57 9:57 10:57 11:57 12:57
1:58 2:58 3:58 4:58 5:58 6:58 7:58 8:58 9:58 10:58 11:58 12:58
1:59 2:59 3:59 4:59 5:59 6:59 7:59 8:59 9:59 10:59 11:59 12:59

```

## 연습문제 6-4

반지름이 1인 원 안에 다트를 던져서 원주율 구하기. 설명은 다음장에



### 6-4 설명

survivalcoding.com

1. 던질 횟수를 입력해주세요 를 출력한다
2. 키보드로부터 long값을 변수 tries 에 입력 받는다
3. 정수형 hits 변수를 0으로 초기화 한다
4. 입력 받은 tries 의 수 만큼 for 문을 반복하며 아래 a, b 를 수행한다
  - a. 다트가 꽂히는 좌표 x, y 를 랜덤한 값으로 정해되 범위는 -1 ~ 1 사이의 실수(double) 로 한다
    - i. 힌트 : new Random().nextDouble() 는 0 ~ 1 사이의 실수를 랜덤하게 리턴 해 준다
  - b. 다트가 꽂힌 좌표가 원 안에 있을 경우 hits 를 증가연산자를 사용하여 1 증가 시킨다
    - i. 힌트 : 두 점 (x1, y1), (x2, y2) 의 거리는  $\text{Math.sqrt}((x1 - x2) * (x1 - x2) + (y1 - y2) * (y1 - y2))$  로 구할 수 있음
5. 반복이 끝나면 실수형 변수 piEstimate 를 선언과 동시에 PI 값을 계산하여 대입하여 초기화 한다
  - a. 힌트 : hits / tries 의 비율은 원의 면적 / 사각형의 면적과 같고 이는 원주율(Pi) / 4 와도 같다. 이 관계를 이용하여 PI를 구하면 됨  
식은 이겁니다 :  $\text{piEstimate} = 4 * \text{hits} / \text{tries}$
6. 마지막에 PI 값의 예상값 piEstimate 를 출력한다.
7. 3.141592.... 에 가까운 값이 나오는지 확인한다.

## 6.4 에 대한 설명 - 수학적 배경지식

### 확률론 - 기하적 확률과 통계적 확률

이 문제는 기하적 확률과 통계적 확률을 접목하여  $\pi$ 의 값을 구하는 문제이다.

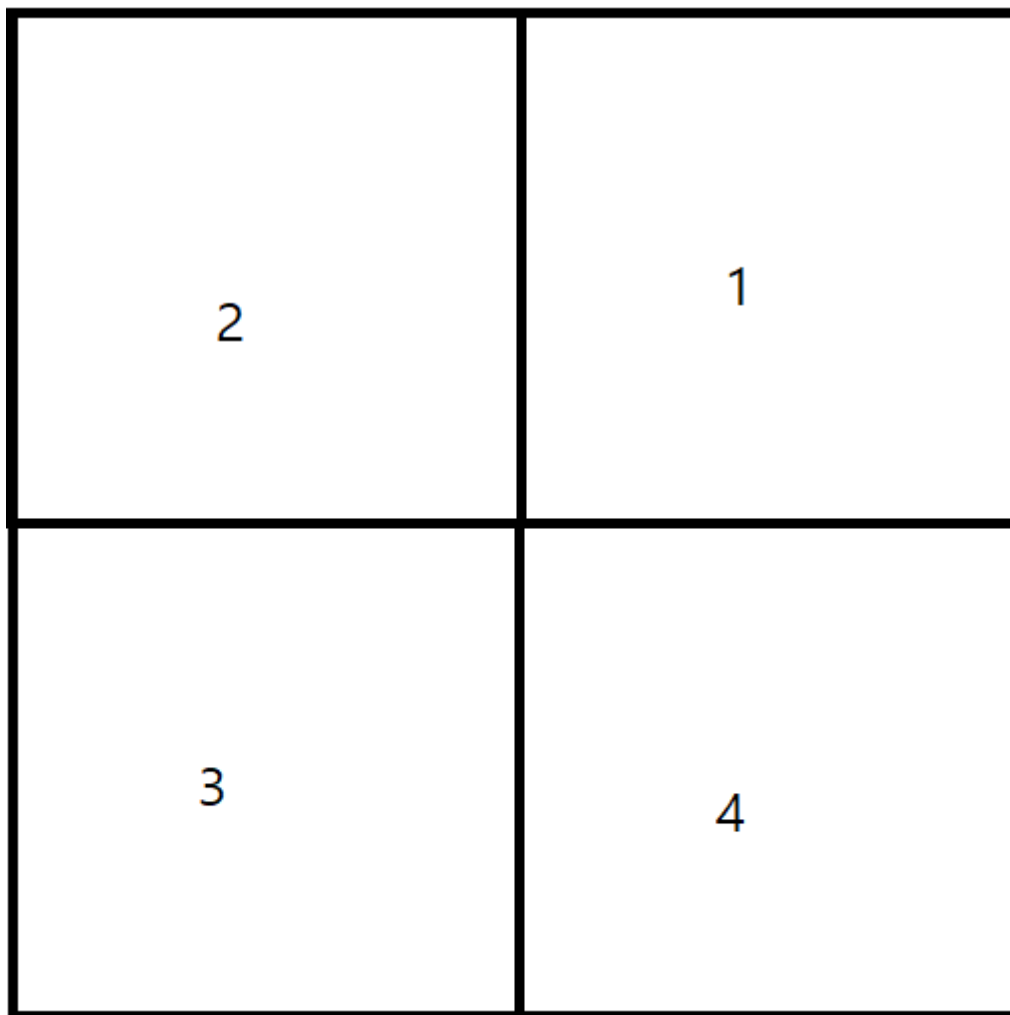
수학에서의 확률은 크게 3가지 종류가 있다.

이론적인 사고를 통해 도출하는 수학적 확률

실험의 결과를 통해 도출하는 통계적 확률

도형의 넓이 비율을 통해 도출하는 기하적 확률

예를 들어 이를 설명해보자.



위와 같이 4분할 된 한 변의 길이가 10인 정사각형에 돌을 던져 1이 나올 확률을 구한다고 하자.

수학적 확률 : 4분할 된 사각형 4개 중 1번 사각형은 1개이므로  $1/4$

통계적 확률 : 돌을 10000번 던져 1번 사각형에 맞은 횟수가 2500번 이므로,  $2500/10000 = 1/4$

기하적 확률 : 1번 정사각형의 넓이 25 / 전체 정사각형 넓이 100 =  $25/100 = 1/4$

## 큰 수의 법칙

통계적 확률의 성질 중 하나이다. 시행횟수가 많아질 수록 통계적 확률의 정확도가 높아진다는 뜻이다.

즉, 시행 횟수가 많아질수록 통계적 확률은 이론값 혹은 수학적확률(기하적확률)에 근사해져는 것이다.

단순히 생각해서 위의 예시에서 10000번 던져 나온 정확도가 100번 던져 나온 정확도 보다 정확하고, 신뢰 가능하다는 뜻이다.

## 원의 넓이 공식

원의 정의를 이용하면 원의 넓이 공식을 구할 수 있다.

원의 정의 : 한 점(중점)에서 떨어진 거리(반지름)가 같은 점들의 모임

즉, 두 점사이 거리 공식을 이용하면 원의 넓이 공식을 구할 수 있다.

두 점사이 거리 공식은 문제의 힌트에서 주어졌다.

$$distance = \sqrt{(x1 - x2)^2 + (y1 - y2)^2}$$

이때 두 점사이 거리가 반지름이므로 , 양변에 제곱을 해주면 원의 넓이 공식이 된다.

$$distance = r$$

$$r^2 = (x1 - x2)^2 + (y1 - y2)^2$$

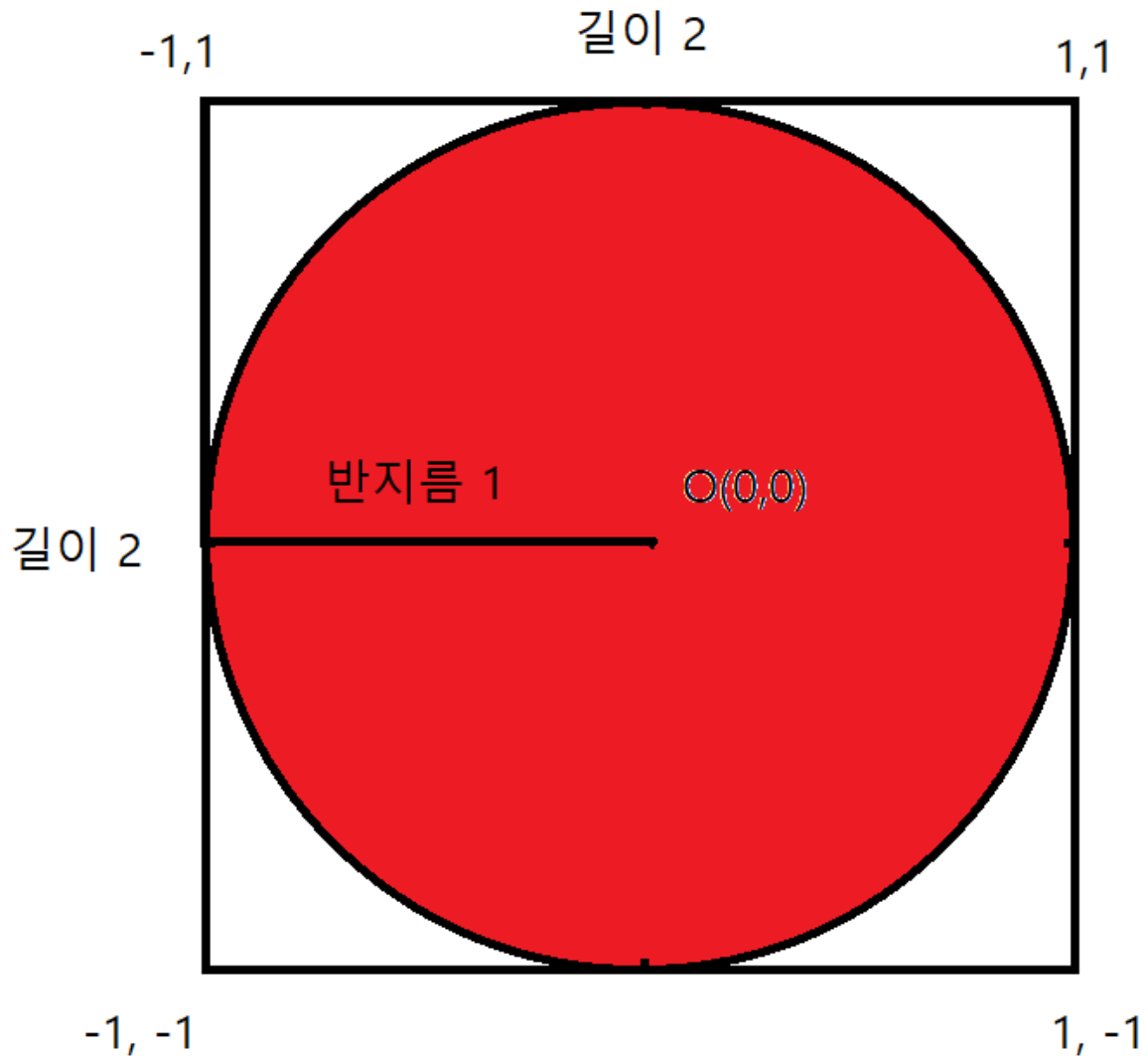
원의 넓이 공식에서 (x2, y2)은 중점의 좌표이다. 이 문제의 경우 중점의 좌표가 (0,0)이므로 식은 다음과 같다.

$$r^2 = (x)^2 + (y)^2$$

마지막으로, 우리가 원하는 부분은 원의 테두리가 아닌, 원의 내부이므로, 부등식으로 나타낸다.

$$(x)^2 + (y)^2 \leq r^2$$

## 기하적 확률에 대한 추가 설명



이 문제를 푸는 가장 중요한 전제는 기하적 확률과 통계적 확률이 같을 것이라는 것이다.

일단 시행횟수인 tries가 -1 ~ 1의 범위에 있으므로, 사각형에는 무조건 맞는다.

hits는 원에 맞은 경우의 수 이다.

통계적 확률은 다트가 원에 맞은 횟수 / 다트가 사각형이내에 맞은 횟수(시행횟수)이다.

이를 식으로 나타내어 정리하면 다음과 같다.

$$P1 = hits/tries$$

기하적 확률은 원의 넓이 / 정사각형의 넓이 이다.

이를 식으로 나타내어 정리하면 다음과 같다.

$$P2 = PI * r^2 / x^2$$

$$P2 = PI * 1^2 / 2^2$$

$$P2 = PI/4$$

대전제에 의하여,  $P1 = P2$  이므로 이를 정리하면 다음과 같다.

$$P2 = P1$$

$$PI/4 = hits/tries$$

$$PI = hits/tries * 4$$

위의 과정과 같이 PI값의 추정값을 구할 수 있다.

이는 실제 PI 값이 아닌 추정값이므로, piEstimate로 표현하였다.

```
package com.example;

import java.util.*;
import java.math.*;

public class Main {
    public static void main(String[] args) throws Exception {
        Scanner scanner = new Scanner(System.in);
        Random random = new Random();
        double PI = Math.PI;
        System.out.println("던질 횟수를 입력해 주세요");
        long tries = scanner.nextInt();
        int hits = 0;

        for (int k = 0; k < tries; k++) {
            double x = 2 * random.nextDouble() - 1;
            double y = 2 * random.nextDouble() - 1;

            if (Math.pow(x, 2) + Math.pow(y, 2) <= 1) {
                hits++;
            }
        }
        double piEstimate = (4 * hits) / (double) tries;
        double error = Math.abs(1 - piEstimate / PI) * 100;
        System.out.println("PI의 이론값 : " + PI);
        System.out.println("PI의 실험값 : " + piEstimate);
        System.out.println("오차율 : " + error + "%");
    }
}
```



```

Problems @ Javadoc Declaration Console ×
<terminated> Main [Java Application] C:\Users\admin\W
던질 횟수를 입력해 주세요
5
PI의 이론값 : 3.141592653589793
PI의 실험값 : 1.6
오차율 : 49.070418210593495%

```

```

Problems @ Javadoc Declaration Console ×
<terminated> Main [Java Application] C:\Users\admin\W
던질 횟수를 입력해 주세요
1600
PI의 이론값 : 3.141592653589793
PI의 실험값 : 3.2075
오차율 : 2.097895993450871%

```

```

Problems @ Javadoc Declaration Console ×
<terminated> Main [Java Application] C:\Users\admin\W
던질 횟수를 입력해 주세요
82500
PI의 이론값 : 3.141592653589793
PI의 실험값 : 3.1365333333333334
오차율 : 0.16104316550010767%

```

```

Problems @ Javadoc Declaration Console ×
<terminated> Main [Java Application] C:\Users\admin\W
던질 횟수를 입력해 주세요
4561230
PI의 이론값 : 3.141592653589793
PI의 실험값 : 3.140938738015842
오차율 : 0.020814779191813404%

```

큰 수의 법칙에 의해 시도 횟수와 정확도가 비례함을 알 수 있다.

## 6.4 번 문제 풀이

# 주말과제

편하게 풀어

```

package com.example;

import java.util.*;

public class Main {
    public static void main(String[] args) throws Exception {
        Scanner scanner = new Scanner(System.in);
        String[] inputs = scanner.nextLine().split(" ");
        int a = Integer.parseInt(inputs[0]);
        int b = Integer.parseInt(inputs[1]);
        int R = Integer.parseInt(inputs[2]);
        int N = Integer.parseInt(scanner.nextLine());
        String[] trees = new String[N];

        for (int k = 0; k < N; k++) {
            String[] tmp = scanner.nextLine().split(" ");
            int x = Integer.parseInt(tmp[0]);
            int y = Integer.parseInt(tmp[1]);
            if (Math.pow((x - a), 2) + Math.pow((y - b), 2) <= Math.pow(R, 2)) {
                trees[k] = "noisy";
            } else {
                trees[k] = "silent";
            }
        }
        for (String tree : trees) {

```

```

        System.out.println(tree);
    }
}

```

## 변태화 이후

```

package com.example;

import java.util.*;

public class Main {
    public static void main(String[] args) throws Exception {
        Scanner scanner = new Scanner(System.in);
        int[] firstline = toInteger(scanner);
        int[] secondline = toInteger(scanner);
        int[] inputs = zipIntArrays(firstline, secondline);
        String[] trees = solution(scanner, inputs);
        showOutputs(trees);
    }

    public static int[] toInteger(Scanner scanner) {
        String[] inputs = scanner.nextLine().split(" ");
        int[] outputs = new int[inputs.length];
        for (int i = 0; i < inputs.length; i++) {
            outputs[i] = Integer.parseInt(inputs[i]);
        }
        return outputs;
    }

    public static int[] zipIntArrays(int[] array1, int[] array2) {
        int[] result = new int[array1.length + array2.length];
        for (int i = 0; i < result.length; i++) {
            if (i < array1.length) {
                result[i] = array1[i];
            } else {
                result[i] = array2[i - array1.length];
            }
        }
        return result;
    }

    public static String[] solution(Scanner scanner, int[] inputs) {
        int xOfFocus = inputs[0];
        int yOfFocus = inputs[1];
        int radius = inputs[2];
        int number = inputs[3];
        String[] result = new String[number];
        for (int k = 0; k < number; k++) {
            int[] newInputs = toInteger(scanner);
            int xOfTree = newInputs[0];
            int yOfTree = newInputs[1];
            result[k] = determine(xOfTree, yOfTree, xOfFocus, yOfFocus, radius);
        }
        return result;
    }
}

```

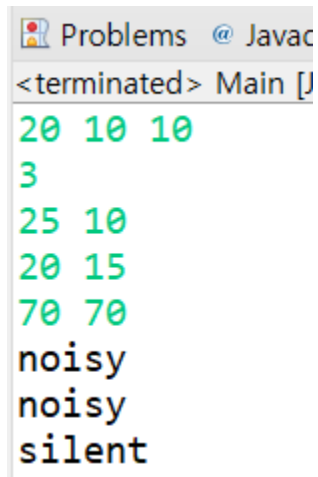
```

    }

    public static String determine(int xOfTree, int yOfTree, int xOfFocus, int yOfFocus,
        int radius) {
        String result = "silent";
        if (Math.pow((xOfTree - xOfFocus), 2) + Math.pow((yOfTree - yOfFocus), 2) <= Math
            .pow(radius, 2)) {
            result = "noisy";
        }
        return result;
    }

    public static void showOutputs(String[] outputs) {
        for (String output : outputs) {
            System.out.println(output);
        }
    }
}

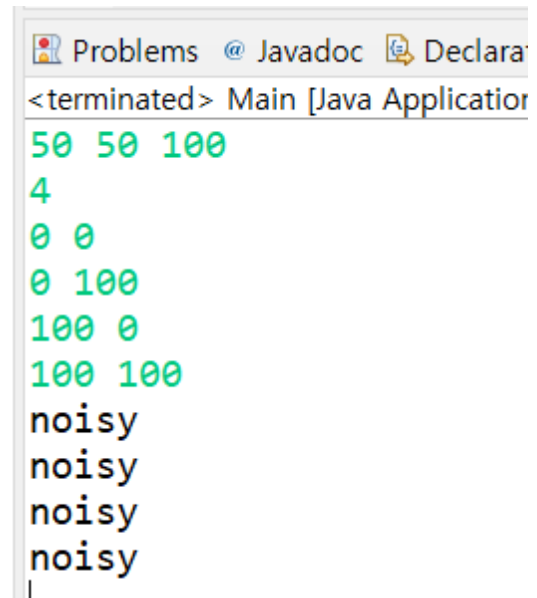
```



```

Problems @ Javac
<terminated> Main [J
20 10 10
3
25 10
20 15
70 70
noisy
noisy
silent

```



```

Problems @ Javadoc Declara
<terminated> Main [Java Application
50 50 100
4
0 0
0 100
100 0
100 100
noisy
noisy
noisy
noisy

```