StringTokenizer

문자열이 특정 구분자로 연결되어 있을 경우, 구분자를 기준으로 부분 문자열을 분리하기 위해서는 'String의 split()' 메소드를 이용하거나 java.util 패키지의 'StringTokenizer' 클래스를 이용할 수 있다.

split() 메소드는 정규 표현식으로 구분하고, StringTokenizer는 문자로 구분한다는 차이점이 있다.

String 클래스의 split() 메소드는 정규 표현식을 구분자로 해서 문자열을 분리한 후, 배열에 저장하고 리턴한다.

파이프(|) 기호로 연결한 정규 표현식을 매개값으로 제공하면, split() 메소드는 이 기호들을 구분자로 해서 부분 문자열을 추출한다.

```
// StringTokenizer 는 특수한 기준자를 바탕으로 문자열을 분리한다.
* StringTokenizer st = new StringTokenizer(received, "#");
String recipient = st.nextToken();
String msg2Send = st.nextToken();*/

StringTokenizer st = new StringTokenizer(received, delim: "-");
String recipient = st.nextToken();
String msg2Send = st.nextToken();
```

실험대상-생쥐
recipient:실험대상 "-" msg2send:생쥐

```
public static void main(String[] args){
    String text = "홍길동&김코딩,강자바,이순신-오로라";

    String[] names = text.split("&|,|-");

    for(String name : names) {
        System.out.println(name);
    }
}
```

Problems @ Jav
<terminated> New
홍길동
김코딩
강자바
이순신
오로라

split을 이용한것으로 &,- 을 |(정규표현식)으로 문자열을 분리한다.

```
for(int i = 0; i < MAXTHREAD; i++) {
    thr[i].join();
}

PerformanceUtil.performanceCheckEnd();
PerformanceUtil.printPerformance();
```

thr[i].join -> join으로 다른 스레드의 종료를 기다리게 해줌.
과정을 완료 후 performanceCheckEnd와 PrintPerformance를 실행하도록 만듬.