

□ 개념 확인

(1) **[생성형 AI 활용]** 스레드 객체를 생성하는 방법에 대하여 예와 함께 설명하세요

생성형 AI	perplexity
질문	스레드 객체를 생성하는 방법에 대하여 예와 함께 설명하세요
답변	<p>스레드 객체를 생성하는 방법과 예제</p> <p>자바에서 스레드 객체를 생성하는 방법은 대표적으로 두 가지가 있습니다. 각각의 방법과 예제를 아래에 설명합니다.</p> <h3>1. Thread 클래스를 상속받아 생성</h3> <ul style="list-style-type: none">Thread 클래스를 상속받아 새로운 클래스를 정의하고, run() 메서드를 오버라이드하여 스레드가 실행할 코드를 작성합니다.정의한 클래스로 객체를 생성한 뒤, start() 메서드를 호출하여 스레드를 실행합니다. <pre>java class MyThread extends Thread { @Override public void run() { System.out.println("Thread 실행: " + this.getName()); } } public class Main { public static void main(String[] args) { MyThread thread = new MyThread(); // 스레드 객체 생성 thread.start(); // 스레드 실행 } }</pre> <p>이 방식은 스레드 클래스를 직접 정의할 수 있으나, 이미 다른 클래스를 상속받고 있다면 사용할 수 없습니다 17.</p> <h3>2. Runnable 인터페이스를 구현하여 생성</h3> <ul style="list-style-type: none">Runnable 인터페이스를 구현하는 클래스를 정의하고, run() 메서드를 오버라이드합니다.Runnable 구현 객체를 Thread 생성자의 인자로 전달하여 Thread 객체를 생성한 후, start() 메서드로 실행합니다. <pre>java class MyRunnable implements Runnable { @Override public void run() {</pre>

```

        System.out.println("Runnable 실행: " + Thread.currentThread().getName());
    }
}

public class Main {
    public static void main(String[] args) {
        Runnable runnable = new MyRunnable(); // Runnable 객체 생성
        Thread thread = new Thread(runnable); // Thread 객체 생성
        thread.start(); // 스레드 실행
    }
}

```

이 방식은 다중 상속이 불가능한 자바에서 더 유연하게 사용할 수 있습니다. 이미 다른 클래스를 상속받고 있는 경우에도 스레드를 구현할 수 있습니다 [126](#).

3. 익명 클래스(Anonymous Class) 또는 람다식 사용

- 간단한 작업의 경우, 익명 클래스를 사용하거나 자바 8 이상에서는 람다식으로도 스레드를 생성할 수 있습니다.

```

java
Thread thread = new Thread(new Runnable() {
    @Override
    public void run() {
        System.out.println("익명 Runnable 실행");
    }
});
thread.start();

```

또는,

```

java
Thread thread = new Thread(() -> System.out.println("람다식 Runnable 실행"));
thread.start();

```

이 방법은 코드가 간결해지는 장점이 있습니다 [6](#).

요약

방법	특징	예제 코드 요약
Thread 클래스 상속	run() 오버라이드, 직접 객체 생성	new MyThread().start();
Runnable 인터페이스 구현	run() 오버라이드, Thread 생성자에 전달	new Thread(new MyRunnable()).start();
익명 클래스/람다식	코드 간결, 일회성 작업에 적합	new Thread(() -> ...).start();

이처럼 자바에서 스레드 객체를 생성하는 주요 방법은 Thread 클래스 상속과 Runnable 인터페이스 구현이 있으며, 각각의 상황에 따라 적절한 방식을 선택하여 사용할 수 있습니다 [1267](#).

(2) [생성형 AI 활용] 문자열 분리 방법에 대하여 예와 함께 설명하세요

생성형 AI	perplexity
질문	문자열 분리 방법에 대하여 예와 함께 설명하세요
답변	<p>문자열 분리 방법과 예제</p> <p>자바에서 문자열을 분리하는 방법은 여러 가지가 있으며, 대표적으로 <code>split()</code> 메서드, <code>substring()</code> 메서드, 그리고 <code>StringTokenizer</code> 클래스를 사용할 수 있습니다. 각 방법과 예제는 다음과 같습니다.</p> <p>1. <code>split()</code> 메서드 사용</p> <ul style="list-style-type: none"><code>split()</code>은 지정한 구분자(문자, 문자열, 정규식 등)를 기준으로 문자열을 분리하여 문자열 배열로 반환합니다.다양한 구분자와 정규식을 사용할 수 있어 가장 많이 활용되는 방법입니다. <p>java</p> <pre>String str = "Apple,Amazon,Google,Microsoft"; String[] result = str.split(","); System.out.println(Arrays.toString(result)); // 출력: [Apple, Amazon, Google, Microsoft]</pre> <p>공백을 기준으로 분리:</p> <p>java</p> <pre>String str = "Apple Amazon Google Microsoft"; String[] result = str.split(" "); System.out.println(Arrays.toString(result)); // 출력: [Apple, Amazon, Google, Microsoft]</pre> <p>정규식을 이용한 복수 구분자 분리:</p> <p>java</p> <pre>String str = "Hello! World. This,is~a test of the split method"; String[] result = str.split("[!.,~]+"); for (String s : result) System.out.println(s.trim()); /* 출력: Hello World This is a test of the split method */</pre> <p>특정 단어로 분리:</p> <p>java</p>

```
String str = "I like apples and I like oranges but I don't like bananas";
String[] result = str.split("and|but");
for (String s : result) System.out.println(s.trim());
/*
출력:
I like apples
I like oranges
I don't like bananas
*/
```

2. substring() 메서드 사용

- substring()은 문자열의 특정 부분을 잘라낼 때 사용합니다.
- 구분자가 일정한 위치에 있을 때 유용합니다.

```
java
String str = "HelloWorld";
String sub = str.substring(0, 5); // 0 번 인덱스부터 5 번 인덱스 전까지
System.out.println(sub);
// 출력: Hello
```

3. StringTokenizer 사용

- StringTokenizer 는 여러 구분자를 지정하여 문자열을 분리할 수 있는 클래스입니다.
- 반복적으로 토큰을 추출할 때 유용합니다.

```
java
import java.util.StringTokenizer;

String str = "Apple,Amazon,Google,Microsoft";
StringTokenizer st = new StringTokenizer(str, ",");
while(st.hasMoreTokens()) {
    System.out.println(st.nextToken());
}
/*
출력:
Apple
Amazon
Google
Microsoft
*/
```

요약

방법	특징	예시 코드 요약
split()	구분자/정규식 기준, 배열 반환	str.split(",")
substring()	인덱스 범위로 부분 문자열 추출	str.substring(0, 5)
StringTokenizer	여러 구분자, 반복적 토큰 추출	new StringTokenizer(str, ",")

이처럼 자바에서는 상황에 따라 다양한 문자열 분리 방법을 사용할 수 있습니다.

- (3) 정수 문자열을 입력 받은 후 소수의 합을 계산하여 출력 하세요. 문자열 분리는 split() 메소드를 사용합니다. 단, 구분자로 콤마를 사용하며 제시된 main()을 사용합니다.

```
public static void main(String[] args) {
    Scanner key = new Scanner(System.in);
    Calc cal=null;

    System.out.print("정수 문자열을 입력하세요. \n 정수와 정수 사이는 ,(콤마)로 구분하세요 \n>> ");
    String digit_str = key.nextLine();
    cal=new Calc(digit_str);
    cal.show();
    System.out.printf("sum = %d\n", cal.sum());
}
```

```
정수와 정수 사이는 ,(콤마)로 구분하세요
>> 11,31,24,53,35,78,49,55,36,96,374,35
11
31
24
53
35
78
49
55
36
96
374
35
소수 : 11 31 53
sum = 95
```

[프로그램 소스]

```
package practice3;

import java.util.Arrays;

public class Calc {
    private String[] arr;
    private int[] nums;
    public Calc(String digitStr) {
        arr = digitStr.split(",");
        nums = new int[arr.length];
        for (int i = 0; i < arr.length; i++) {
            arr[i] = arr[i].trim();
            nums[i] = Integer.parseInt(arr[i].trim());
        }
    }

    public void show() {
        for (int i = 0; i < arr.length; i++) {
            System.out.println(arr[i]);
        }
        System.out.print("소수 : ");
        for (int i = 0; i < nums.length; i++) {
            if(isPrime(nums[i])) {
                System.out.print(nums[i] + " ");
            }
        }
        System.out.println("");
    }

    public boolean isPrime(int n) {
        if (n <= 1) { // 1 이하면 소수가 아님
```

```

        return false;
    }
    for (int i = 2; i < n; i++) {
        if (n % i == 0) { // 약수가 있으면 소수가 아님
            return false;
        }
    }
    return true;
}

public int sum() {
    int sum = 0;
    for (int i = 0; i < nums.length; i++) {
        if (isPrime(nums[i])) {
            sum += nums[i];
        }
    }
    return sum;
}
}

```

```

package practice3;

import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        Scanner key = new Scanner(System.in);
        Calc cal = null; // 11, 31, 24, 53, 35, 78, 49, 55, 36, 96, 374, 35

        System.out.print("정수 문자열을 입력하세요. \n 정수와 정수 사이는 , (콤마)로 구분하세요
\n>> ");
        String digit_str = key.nextLine();
        cal = new Calc(digit_str);
        cal.show();
        System.out.printf("sum = %d\n", cal.sum());
    }
}

```

[실행 결과]

```
실행 Main x
/Users/hataeyeong/Library/Java/JavaVirtualMachines/
정수 문자열을 입력하세요.
정수와 정수 사이는 ,(coma)로 구분하세요
>> 11, 31, 24, 53, 35, 78, 49, 55, 36, 96, 374, 35
11
31
24
53
35
78
49
55
36
96
374
35
소수 : 11 31 53
sum = 95

종료 코드 0(으)로 완료된 프로세스
```

(4) 다음과 같이 특수 문자와 영문자, 숫자를 출력하는 스레드를 작성하고 테스트 하는 프로그램을 작성하세요. 단, 출력시 0.5 초의 간격이 있도록 합니다. 특수 문자와 영문자, 숫자 구간은 본인이 임의로 결정합니다. 스레드 객체는 람다식, 익명자식객체, 익명구현객체를 각각 사용합니다.

- 특수문자: ◆
- 숫자: 1
- 영문자: F
- 영문자: G
- 숫자: 2
- 특수문자: ▣
- 숫자: 3
- 영문자: H
- 특수문자: ※
- 숫자: 4
- 영문자: I
- 특수문자: ○
- 영문자: J
- 특수문자: ♠
- 숫자: 5
- 특수문자: ★
- 영문자: K
- 특수문자: ♣

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

    }
}
```

[프로그램 소스]

```
package practice4;

public class TestThread {
    public static void main(String[] args) {
        Thread t1 = new Thread(() -> {
            try {
```

```

        String[] strs = {"◆", "■", "※", "○", "♠", "★", "♣"};
        for (int i = 0; i < strs.length; i++) {
            System.out.println("특수문자 : " + strs[i]);
            Thread.sleep(500);
        }
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
}

});

Thread t2 = new Thread() {
    @Override
    public void run() {
        try {
            int[] nums = {1, 2, 3, 4, 5};
            for (int i = 0; i < nums.length; i++) {
                System.out.println("숫자 : " + nums[i]);
                Thread.sleep(500);
            }
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
};

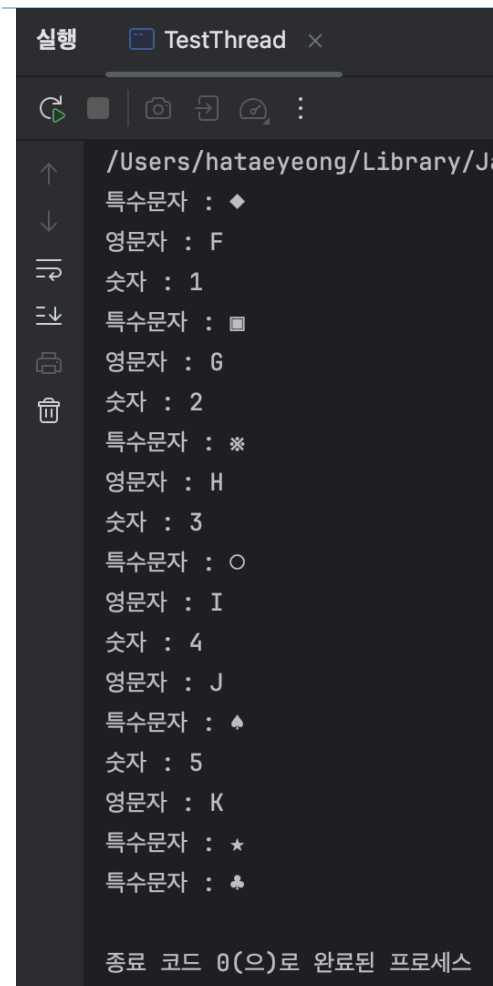
Thread t3 = new Thread(new Runnable() {
    @Override
    public void run() {
        try {
            String[] strs = {"F", "G", "H", "I", "J", "K"};
            for (int i = 0; i < strs.length; i++) {
                System.out.println("영문자 : " + strs[i]);
                Thread.sleep(500);
            }
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
});

t1.start();
t2.start();
t3.start();

}
}

```

[실행 결과]



(5) 다음과 같은 배열 원소를 출력하는 각각의 스레드 객체를 생성하고 실행하는 프로그램을 작성하세요. 단, 상속과 인터페이스 구현 두 가지 방법으로 제시하세요

```
int-thread : 3
string-thread : hallym
string-thread : software
int-thread : 14
string-thread : big
int-thread : 52
string-thread : content
int-thread : 7
string-thread : IoT
int-thread : 11
string-thread : data
int-thread : 8
int-thread : 90
int-thread : 74
```

```
int-thread : 3
string-thread : hallym
int-thread : 14
string-thread : software
int-thread : 52
string-thread : big
int-thread : 7
string-thread : content
int-thread : 11
string-thread : IoT
int-thread : 8
string-thread : data
int-thread : 90
int-thread : 74
```

```
int[] intArray = {3,14,52,7,11,8, 90,74};
```

```
String[] strArray = {"hallym", "software", "big", "content", "IoT", "data"};
```

[프로그램 소스]

```
package practice5;

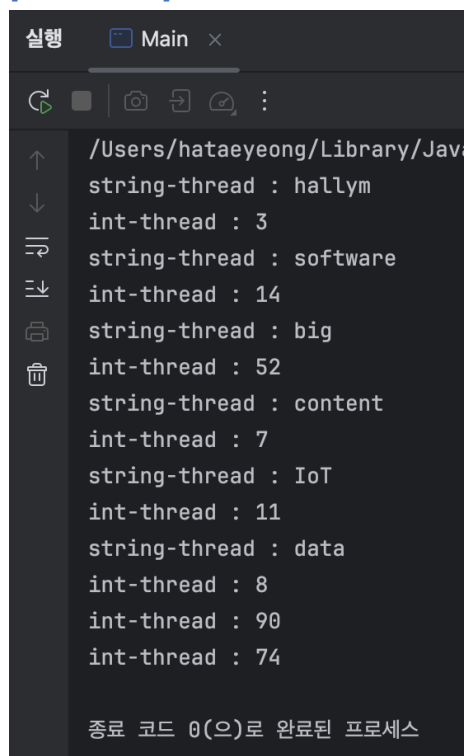
class IntThread extends Thread {
    @Override
    public void run() {
        int[] intArray = {3,14,52,7,11,8,90,74};
        try {
            for (int i = 0; i < intArray.length; i++) {
                System.out.println("int-thread : " + intArray[i]);
                Thread.sleep(1000);
            }
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}
```

```
}  
}
```

```
package practice5;  
  
class StringThread implements Runnable {  
    @Override  
    public void run() {  
        String[] strArray = {"hallym", "software", "big", "content", "IoT", "data"};  
        try {  
            for (int i = 0; i < strArray.length; i++) {  
                System.out.println("string-thread : " + strArray[i]);  
                Thread.sleep(1000);  
            }  
        } catch (InterruptedException e) {  
            e.printStackTrace();  
        }  
    }  
}
```

```
package practice5;  
  
public class Main {  
    public static void main(String[] args) {  
        Thread t1 = new IntThread();  
        Runnable r = new StringThread();  
        Thread t2 = new Thread(r);  
        t1.start();  
        t2.start();  
    }  
}
```

[실행 결과]



```
실행 Main x  
/Users/hataeyeong/Library/Jav  
string-thread : hallym  
int-thread : 3  
string-thread : software  
int-thread : 14  
string-thread : big  
int-thread : 52  
string-thread : content  
int-thread : 7  
string-thread : IoT  
int-thread : 11  
string-thread : data  
int-thread : 8  
int-thread : 90  
int-thread : 74  
  
종료 코드 0(으)로 완료된 프로세스
```

응용 프로그래밍

- (6) 현재 날짜와 시간을 출력한 후, 현재를 기준으로 5 일후의 날짜도 함께 출력하세요. 단, time 패키지 사용

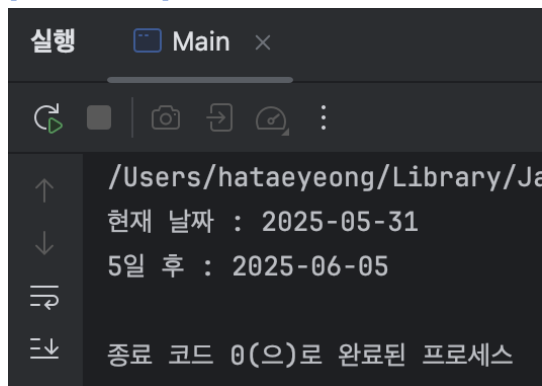
[프로그램 소스]

```
package practice6;

import java.time.LocalDate;

public class Main {
    public static void main(String[] args) {
        LocalDate today = LocalDate.now();
        LocalDate after5days = today.plusDays(5);
        System.out.println("현재 날짜 : " + today);
        System.out.println("5 일 후 : " + after5days);
    }
}
```

[실행 결과]



- (7) 제시된 main() 메소드와 결과를 참고하여 다음 설명에 해당하는 프로그램을 작성하세요.

1. Thread 를 상속받은 Job 클래스를 작성하세요.

>> 필드(private String name, int num)

>> 생성자(이름을 전달하여 저장)

>> run() 메소드 : 0.5 초간 sleep() 후 getNum()메소드 호출, getNum()에서 리턴 받은 값이 10 이면 반복 탈출

>> getNum() : 한 순간에 하나의 스레드만 접근할 수 있는 메소드(임계영역), num 값을 1 증가 시킨 후 증가된 num 값을 리턴

```
public class Test{
    public static void main(String[] args) {
        Thread t1 = new Job("ThreadA");
        Thread t2 = new Job("ThreadB");
        t1.start();
        t2.start();
    }
}
```

```
ThreadB: 1
ThreadA: 1
ThreadB: 2
ThreadA: 2
ThreadB: 3
ThreadA: 3
ThreadB: 4
ThreadA: 4
ThreadB: 5
ThreadA: 5
ThreadB: 6
ThreadA: 6
ThreadB: 7
ThreadA: 7
ThreadB: 8
ThreadA: 8
ThreadB: 9
ThreadA: 9
```

[프로그램 소스]

```
package practice7;

public class Job extends Thread {
    // 필드
    private String name;
    private int num = 0;

    // 생성자
    public Job(String name) {
        this.name = name;
    }

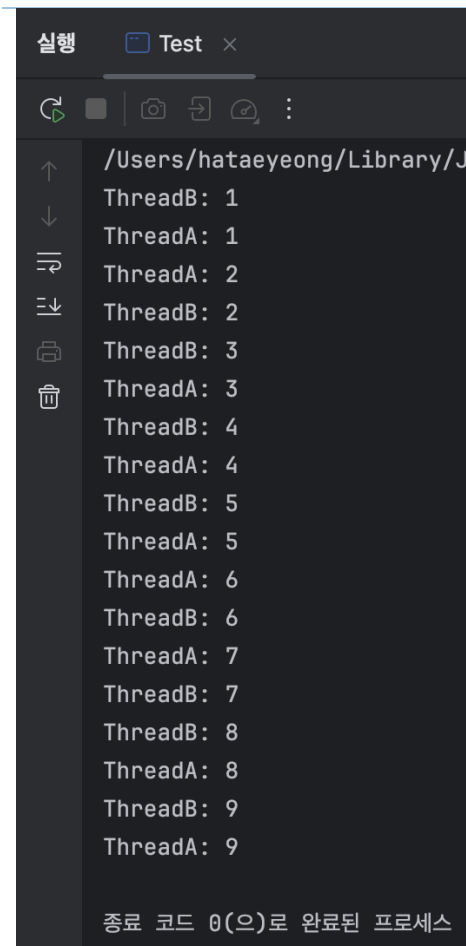
    @Override
    public void run() {
        while (true) {
            try {
                Thread.sleep(500);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            int value = getNum();
            System.out.println(name + ": " + value);
            if (value >= 9) break;
        }
    }

    public synchronized int getNum() {
        num++;
        return num;
    }
}
```

```
package practice7;

public class Test{
    public static void main(String[] args) {
        Thread t1 = new Job("ThreadA");
        Thread t2 = new Job("ThreadB");
        t1.start();
        t2.start();
    }
}
```

[실행 결과]



(8) 11 장 수업자료 31 번 슬라이드에 제시된 예제 프로그램을 다음과 같이 동작될 수 있도록 수정하세요

```
public class WaitNotifyExam {  
    public static void main(String[] args) {  
        MessageBox dataBox = new MessageBox();  
        String[] message = new String[] { "hallym", "big data", "software", "AI" };  
  
        SendThread sendThread = new SendThread(dataBox, message);  
        ReceiveThread receiveThread = new ReceiveThread(dataBox, message.length);  
  
        sendThread.start();  
        receiveThread.start();  
    }  
}
```

SendThread 스레드가 전송한 데이터: hallym
ReceiveThread 스레드가 수신한 데이터: hallym
SendThread 스레드가 전송한 데이터: big data
ReceiveThread 스레드가 수신한 데이터: big data
SendThread 스레드가 전송한 데이터: software
ReceiveThread 스레드가 수신한 데이터: software
SendThread 스레드가 전송한 데이터: AI
ReceiveThread 스레드가 수신한 데이터: AI

[프로그램 소스]

```
package practice8;  
  
public class MessageBox { // 데이터를 임시로 저장하는 공유 객체  
    private String data;
```

```

public synchronized String getData() {
    while (this.data == null) { // 데이터가 없으면 대기
        try { wait(); } catch (InterruptedException e) {}
    }
    String returnValue = data; // 데이터 가져오기
    System.out.println("ReceiveThread 스레드가 수신한 데이터: " + returnValue);
    data = null; // 데이터 초기화
    notify(); // 대기중인 스레드 깨우기
    return returnValue;
}

public synchronized void setData(String data) {
    while (this.data != null) { // 데이터가 있으면 대기
        try { wait(); } catch (InterruptedException e) {}
    }
    this.data = data; // 데이터 저장하기
    System.out.println("SendThread 스레드가 전송한 데이터: " + data);
    notify();
}
}

```

```

package practice8;

public class SendThread extends Thread { // 데이터를 MessageBox 에 순차적으로 넣는 생산자
    역할
    private MessageBox dataBox ;
    private String[] messages;

    public SendThread(MessageBox dataBox, String[] messages) {
        this.dataBox = dataBox;
        this.messages = messages;
    }

    @Override
    public void run() {
        for(String data : messages) {
            dataBox.setData(data);
        }
    }
}

```

```

package practice8;

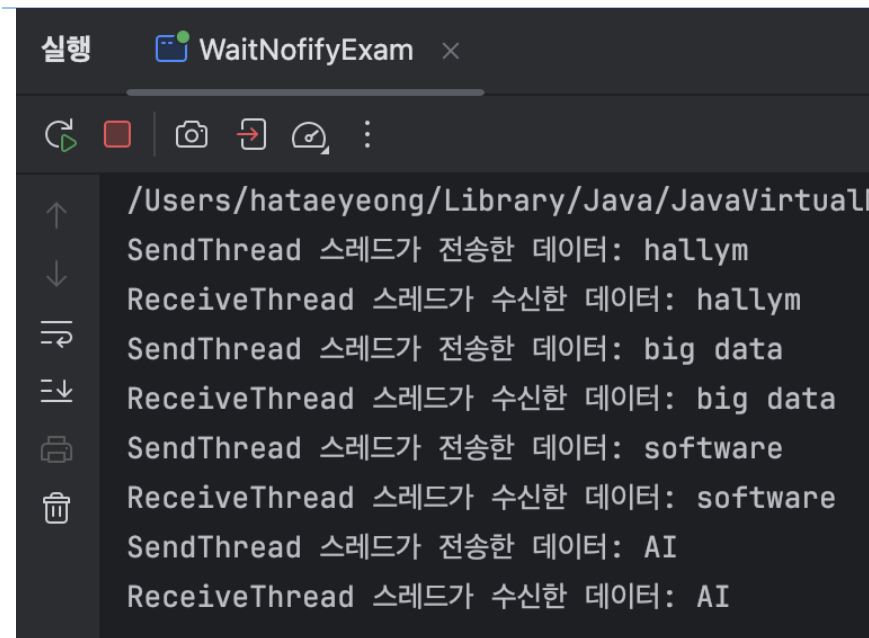
public class ReceiveThread extends Thread { // MessageBox 에서 데이터를 꺼내는 소비자 역할
    private MessageBox dataBox;
    private int count; // ReceiveThread 가 몇 번 데이터를 받아야 하는지를 결정하는 역할

    public ReceiveThread(MessageBox dataBox, int count) {
        this.dataBox = dataBox;
        this.count = count;
    }

    @Override
    public void run() {
        for(int i = 0; i <= count; i++) {
            dataBox.getData();
        }
    }
}

```

[실행 결과]



(9) 다음과 같이 문자열을 처리하는 프로그램을 작성하세요.

- ① StringTokenizer 클래스를 이용하여 사용자로 부터 받은 문자열을 단어로 분리합니다. 단, 공백이 포함된 문자열 입력은 `nextLine()` 사용
- ② 단어들을 문자열 배열에 넣고 이 배열을 Arrays 의 `sort()` 메소드를 이용하여 정렬합니다

```

문자열을 입력하세요 : Did you sleep well last night?
정렬 전 결과 :
Did    you    sleep    well    last    night?
정렬 후 결과 :
Did    last    night?    sleep    well    you

```

[프로그램 소스]

```

package practice9;

import java.util.Arrays;
import java.util.Scanner;
import java.util.StringTokenizer;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("문자열을 입력하세요 : ");
        String str = sc.nextLine(); // Did you sleep well last night?
        System.out.println(str);

        StringTokenizer st = new StringTokenizer(str, " ");

        String[] words = new String[st.countTokens()];

        int i = 0;
        while(st.hasMoreTokens()) {
            words[i++] = st.nextToken();
        }
        System.out.println("정렬 전 결과 :");
        for(String word : words) {
            System.out.print(word + " ");
        }
    }
}

```

```

        System.out.println("");
        Arrays.sort(words);
        System.out.println("정렬 후 결과 :");
        for (String word : words) {
            System.out.print(word + " ");
        }
    }
}

```

[실행결과]

```

실행  WaitNotifyExam x  Main x
/Users/hataeyeong/Library/Java/JavaVirtualMachin
문자열을 입력하세요 : Did you sleep well last night?
Did you sleep well last night?
정렬 전 결과 :
Did you sleep well last night?
정렬 후 결과 :
Did last night? sleep well you
종료 코드 0(으)로 완료된 프로세스

```