|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **[유헬스2\_13주\_실습과제]** | **학번:** | **20175105** | **이름:** | **곽영주** |

* 스레드
* 다중 스레딩(multi-threading)은 하나의 프로그램이 동시에 여러 가지 작업을 할 수 있도록 하는 것
* 각각의 작업은 스레드(thread)라고 불린다
* 스레드를 생성하는 방법
  + java.lang.Thread 클래스를 상속하는 방법
  + java.lang.Runnable 인터페이스를 이용하는 방법

//사용 예1 :Thread 클래스 상속

//숫자를 출력하는 스레드

class DigitThread extends Thread {

//스레드로 처리할 내용을 run()에서 작성

public void run() {

for (int cnt = 0; cnt < 10; cnt++)

System.*out.print(cnt); }}*

public class MultithreadExample1 {

public static void main(String[] args) {

Thread thread = new DigitThread(); // 스레드 객체 생성

thread.start(); // 스레드 시작

for (char ch = 'A'; ch <= 'Z'; ch++)

System.*out.print(ch);*

}

}

//사용 예2: Runnable 인터페이스 구현

class MyRunnable implements Runnable {

String myName;

public MyRunnable(String name) { //스레드를 구분하기 위해 이름 설정

myName = name;}

public void run() {

for (int i = 10; i >= 0; i--)

System.out.print(myName + i + " ");

}}

public class TestThread {

public static void main(String[] args) {

Thread t1 = new Thread( new MyRunnable("A") );

Thread t2 = new Thread( new MyRunnable("B") );

t1.start();

t2.start();}}

|  |
| --- |
| **[실행결과]** |

* 스레드 – 동기화
* critical section이 실행되는 동안 다른 스레드가 공유 데이터를 사용할 수 없도록 만드는 것
* 공유하는 영역에 synchronized 키워드를 추가

class SharedResource {

**long** nF1 = 1, nF2 = 1, nResult = 0;

//임계영역, ***밑줄 친 키워드를 삭제한 후 결과도 확인하여 제출***

**public** **synchronized** **void** changeShared(String szName) {

nResult = nF1 + nF2;

**try**{

nF2 = nF1;

Thread.*sleep*(1000); //시간 지연

nF1 = nResult;

System.*out*.println(szName + ":" + nResult);

}

**catch**(InterruptedException e){}

}

}

class MyThread extends Thread {

SharedResource sharedResource;

String szTName;

**public** MyThread(String szName, SharedResource res) {

**super**(szName);

szTName = szName;

sharedResource = res;

}

**public** **void** run() {

**for**(**int** i = 0;i < 5;i++)

sharedResource.changeShared(szTName);

}

}

public class NotSynchoTest {

public static void main(String args[]) {

SharedResource res = new SharedResource();

MyThread my\_thread1 = new MyThread("thd1", res);

MyThread my\_thread2 = new MyThread("thd2", res);

MyThread my\_thread3 = new MyThread("thd3", res);

my\_thread1.start();

my\_thread2.start();

my\_thread3.start();

}

}

|  |
| --- |
| **[실행결과 – 동기화 이전]**    **[실행결과 – 동기화 이후]** |

* 프로그램 과제

1. 프로그램 과제 생성된 난수 만큼 문자와 숫자를 번갈아 출력하는 스레드 프로그램을 작성하시오. 숫자와 문자를 출력하는 스레드 클래스를 각각 작성. 난수 범위는 1에서 50

import java.util.\*;

public class ThreadCharNum {

public static void main(String[] args) {

//난수를 생성

//숫자와 문자를 출력하는 스레드 객체 각각 생성

//이때 난수를 생성자 형식 매개변수로 전달

//스레드 시작

}

}

public class CharThread { //생성된 난수 만큼 문자 출력하는 스레드

//출력할 문자와 횟수를 저장하기 위한 필드 선언

//매개변수로 문자와 횟수를 받아 필드 초기화하는 생성자

public void run() {

// 횟수만큼 문자를 출력

}

}

public class NumThread { //생성된 난수 만큼 숫자 출력하는 스레드

//출력할 숫자와 횟수를 저장하기 위한 필드 선언

//매개변수로 숫자와 횟수를 받아 필드 초기화하는 생성자

public void run() {

// 횟수만큼 숫자를 출력

}

}

|  |
| --- |
| **[프로그램 소스]**  **import** java.util.\*;  **public** **class** ThreadCharNum {  **public** **static** **void** main(String[] args) {  Random random = **new** Random();  **int** cnt = random.nextInt(50)\*1;  Thread t1 = **new** Thread(**new** NumThread(28,cnt));  Thread t2 = **new** Thread(**new** CharThread('A',cnt));  t1.start();  t2.start();  }  }  **public** **class** CharThread **implements** Runnable {  **char** ch;  **int** cnt;    **public** CharThread(**char** ch, **int** cnt) {  **this**.ch = ch;  **this**.cnt = cnt;  }    **public** **void** run() {  **for**(**int** i=0; i<cnt; i++)  System.***out***.println(ch);  }  }  **public** **class** NumThread **implements** Runnable {  **int** num, cnt;    **public** NumThread(**int** num, **int** cnt) {  **this**.num = num;  **this**.cnt = cnt;  }    **public** **void** run() {  **for**(**int** i=0; i<cnt; i++)  System.***out***.println(num);  }  } |
| **[실행결과]** |

**< 수업자료 ppt 37p ~ 41P 실행결과 >**

**38p**

|  |
| --- |
| **[실행결과]** |

**39p**

|  |
| --- |
| **[실행결과]** |

**40p**

|  |
| --- |
| **[실행결과]** |

**41p**

|  |
| --- |
| **[실행결과]** |