work.com. workshop15 공통 패키지

[문제 1] 다음과 같은 조건을 만족하는 프로그램을 작성 하시오.

- `Thread` 클래스를 상속한 클래스를 정의하고, `run()` 메서드에서 "스레드 실행 중"을 출력하도록 하세요.
- `main` 메서드에서 두 개의 스레드를 생성하고 실행한 뒤, "`main` 종료" 메시지를 출력하세요.

1. 구현 클래스

클래스명	메서드	설명
`MyThread`	`run() : void`	`Thread` 클래스 상속, 작업 내용 작성
`ThreadTest`	`main(String[] args) : void`	스레드 2개 생성 후 실행 및 종료 메시지 출력

2. 실행 결과

출력 예:

```
스레드 실행 중
스레드 실행 중
main 종료
```

[문제 2] 다음과 같은 조건을 만족하는 프로그램을 작성 하시오.

- Thread 클래스를 구현한 Runnable 객체를 이용하여 스레드를 실행하세요.
- Runnable을 구현한 클래스에서 "Runnable 실행 중" 메시지를 출력하세요.

1. 구현 클래스

클래스명	메서드	설명
MyRunnable	run() : void	Runnable 인터페이스 구현
RunnableTest	main(String[] args) : void	Runnable로 스레드 실행

2. 실행 결과

출력 예:

```
Runnable 실행 중
Runnable 실행 중
```

[문제 3] 다음과 같은 조건을 만족하는 프로그램을 작성 하시오.

- 두 개의 스레드가 하나의 공유 자원(카운터)을 증가시키도록 하세요.
- `synchronized` 키워드를 사용하여 동기화 처리하세요.

1. 구현 클래스

클래스명	메서드	설명
`Counter`	`increment() : void` `getCount():int`	공유 자원 증가, `synchronized` 사용
`SyncTest`	`main(String[] args) : void`	스레드 2개에서 카운터 호출

2. 실행 결과

출력 예:

```
최종 카운터 값: 1000
```

3. main() 구조

```
public static void main(String[] args) throws InterruptedException {
        Counter counter = new Counter();
        Runnable task = () -> {
            for (int i = 0; i < 500; i++) {
                counter.increment();
            }
        };
        Thread t1 = new Thread(task);
        Thread t2 = new Thread(task);
        t1.start();
        t2.start();
        t1.join();
        t2.join();

        System.out.println("최종 카운터 값: " + counter.getCount());
    }
```

[문제 4] Thread.sleep()을 사용하여 1초 간격으로 5번 메시지를 출력하세요.

1. 구현 클래스

클래스명	메서드	설명
SleepThread	run() : void	1초 간격 출력 구현
SleepTest	main(String[] args) : void	스레드 시작

2. 실행 결과

출력 예:

```
출력: 1
출력: 2
...
출력: 5
```

3. main() 구조

```
public static void main(String[] args) throws InterruptedException {
       SleepThread st = new SleepThread();
         st.start();
    }
```

[문제 5] Virtual Thread를 사용하여 다중 작업을 병렬로 처리하고, 실행 메시지를 출력하시오

- `Thread.ofVirtual().start(...)` 또는 `Executors.newVirtualThreadPerTaskExecutor()`를 사용하여 가상 스레드를 여러 개 동시에 실행한다
- 각 가상 스레드는 0.5초씩 sleep한 후 "가상 스레드 x번 실행 중" 메시지를 출력한다
- `main` 스레드는 "`main` 종료" 메시지를 마지막에 출력하도록 한다.

1. 구현 클래스

클래스명	메서드	설명
`VirtualTest`	`main(String[] args) : void`	가상 스레드 여러 개 생성 및 병렬 실행 구현

2. 실행 결과

출력 예:

```
가상 스레드 1번 실행 중
가상 스레드 2번 실행 중
가상 스레드 3번 실행 중
가상 스레드 4번 실행 중
가상 스레드 5번 실행 중
main 종료
```

3. main()구조

```
public static void main(String[] args) throws InterruptedException {

 ① Virtual Thread 용 Executor 생성

 ② 5 개의 Virtual Thread 제출

 ③ 스레드 시작 전 0.5 초 대기

 ④ 대기 후 메시지 출력

 ⑤ Executor 종료 요청

 ⑥ 모든 작업이 종료될 때까지 대기

 ⑦ 모든 가상 스레드 종료 후 메시지 출력

}
```

[문제 6] 다음과 같은 조건을 만족하는 프로그램을 작성 하시오.

- CompletableFuture.supplyAsync()를 사용하여 숫자 두 개를 더하는 비동기 작업을 수행한다.
- 결과를 받아서 "계산 결과: X" 형태로 출력한다.
- main()에서는 "main 종료"를 마지막에 출력한다

1. 구현 클래스

클래스명	메서드	설명
AsyncCalcTest	main(String[] args) : void	비동기 계산 및 결과 출력 구현

2. 실행 결과

출력 예:

```
계산 결과: 300
main 종료
```

3. main()구조

```
public static void main(String[] args) throws InterruptedException {
① 비동기 계산 시작 (200 + 100)
② 0.5초간 연산 대기

③ 결과 출력 (계산 결과: 300)

④ main 종료 메시지
 }
```

[문제 7] 다음과 같은 조건을 만족하는 프로그램을 작성 하시오.

- Executors.newFixedThreadPool(3)를 사용하여 스레드 수가 3개인 고정 스레드 풀(FixedThreadPool)을 생성한다.
- 총 **5개의 작업**을 스레드 풀에 등록하여 실행한다.
- 각 작업은 "작업 X 처리 중"이라는 메시지를 출력한다. (X는 작업 번호)
- shutdown()을 호출하여 **더 이상 작업을 추가할 수 없도록 종료 처리**한다

1. 구현 클래스

클래스명	메서드	설명
PoolExample	main(String[] args) : void	스레드풀 생성 및 실행

2. 실행 결과

출력 예: 실행결과 순서는 상관하지 않는다.

```
작업 1 처리 중
...
작업 5 처리 중
```

[문제 8] 다음과 같은 조건을 만족하는 프로그램을 작성 하시오.

- Buffer 클래스를 생성하여 하나의 데이터를 저장할 수 있는 공유 자원을 구현한다.
- synchronized, wait(), notify()를 사용하여 생산자(Producer)와 소비자(Consumer) 간의 데이터 충돌 없이 순차 처리를 구현한다.
- 생산자는 1부터 5까지 숫자를 차례대로 저장하고, 소비자는 저장된 데이터를 차례대로 꺼내 출력한다
- 항상 "생산됨 → 소비됨" 순으로 나타나야 하며, 순서가 바뀌면 동기화 실패이다.

1. 구현 클래스

클래스명	메서드	설명
Buffer	Produce :void Consume: void	synchronized, wait/notify 사용
PCMain	main(String[] args) : void	생산자/소비자 스레드 실행

2. 실행 결과

출력 예:

```
생산됨: 1
소비됨: 1
….
생산됨: 5
소비됨: 5
```

3. main()구조

```
public static void main(String[] args) {
		// ① 공유 자원 생성
		Buffer buffer = new Buffer();
		// ② 생산자 스레드 정의
		Thread producer = new Thread(() -> {
			for (int i = 1; i <= 5; i++)
				buffer.produce(i);
		});
		// ③ 소비자 스레드 정의
		Thread consumer = new Thread(() -> {
			for (int i = 1; i <= 5; i++)
				buffer.consume();
		});
		// ④ 스레드 시작
		producer.start();
		consumer.start();
	}
}
```

[문제 9] 다음과 같은 조건을 만족하는 프로그램을 작성 하시오.

- `ReentrantLock` 클래스를 사용하여 **스레드 간 락 기반 동기화**를 구현한다.
- `safeAccess()` 메서드에서는 락을 획득하고 "잠금 획득" 메시지를 출력한 뒤, 종료 시 반드시 "잠금 해제" 메시지를 출력한다.
- `main` 메서드에서는 두 개의 스레드를 생성하여 `safeAccess()` 메서드에 동시에 접근하도록 한다.

1. 구현 클래스

클래스명	메서드	설명
SharedResource	run(safeAccess() : void	ReentrantLock 사용
LockTest	main(String[] args) : void	스레드 2개를 생성하여 자원에 접근

2. 실행 결과

출력 예:

```
잠금 획득
잠금 해제
잠금 획득
잠금 해제
```

3. main() 구조

```
public static void main(String[] args) {
        // ① 공유 자원 객체 생성
        SharedResource res = new SharedResource();

        // ② 두 개의 스레드 정의 및 실행
        Thread t1 = new Thread(res::safeAccess);
        Thread t2 = new Thread(res::safeAccess);

        // ③ 스레드 실행
        t1.start();
        t2.start();
    }
```

[문제 10] 다음과 같은 조건을 만족하는 프로그램을 작성 하시오.

- CompletableFuture.supplyAsync()를 사용하여 **두 수를 더하는 계산을 비동기적으로 수행**한다.
- 계산된 결과에 후속 작업으로 **2를 곱한 결과를 다시 계산**한다.
- 마지막으로 결과를 출력한 후, "main 종료" 메시지를 출력한다

1. 구현 클래스

클래스명	메서드	설명
FutureExample	main(String[] args) : void	CompletableFuture 사용

2. 실행 결과

출력 예:

```
비동기 계산 시작
최초 계산 결과: 300
후속 처리 결과 (x2): 600
main 종료
```

3. main() 구조

```
public static void main(String[] args) {
        System.out.println("비동기 계산 시작");

        // ① supplyAsync 로 비동기 계산 (200 + 100)

        // ② 후속 작업: 결과에 2 를 곱함

        // ③ 최종 결과 출력

        // ④ 모든 작업 완료까지 대기

}
```