KB금융그룹 | 국민의 평생 금융파트너

2024년 상반기 K-디지털 트레이닝

스트림 요소 처리

[KB] IT's Your Life

KB 국민은행

스트림 요소 처리

스트림을 이용하여 다음과 같이 출력하는 코드를 완성하세요.

```
package ch17.sec01;

import java.util.HashSet;
import java.util.Iterator;
import java.util.Set;
import java.util.stream.Stream;

public class StreamExample {
  public static void main(String[] args) {
    Set<String> set = new HashSet< >();
    set.add("홍길동");
    set.add("신용권");
    set.add("감자바");



  }
}
```

```
홍길동
신용권
감자바
```

스트림 요소 처리

StreamExample.java

```
package ch17.sec01;

import java.util.HashSet;
import java.util.Iterator;
import java.util.Set;
import java.util.stream.Stream;

public class StreamExample {
  public static void main(String[] args) {
    //Set 컬렉션 생성
    Set<String> set = new HashSet< >();
    set.add("홍길동");
    set.add("신용권");
    set.add("감자바");

    //Stream을 이용한 요소 반복 처리
    Stream<String> stream = set.stream();
    stream.forEach( name -> System.out.println(name) );
  }
}
```

```
홍길동
신용권
감자바
```

스트림 요소 처리

다음과 같이 Student 클래스를 작성하세요.

```
package ch17.sec03;

public class Student {
  private String name;
  private int score;

  public Student (String name, int score) {
    this.name = name;
    this.score = score;
  }

  public String getName() { return name; }
  public int getScore() { return score; }
}
```

스트림 요소 처리

- **스트림을 이용하여 다음과 같이 출력되도록 코드를 완성하세요.**
  - 스트림 처리시 메서드 체인닝을 사용함.

```
package ch17.sec03;

import java.util.Arrays;
import java.util.List;

public class StreamPipeLineExample {
  public static void main(String[] args) {
    List<Student> list = Arrays.asList(
        new Student("홍길동", 10),
        new Student("신용권", 20),
        new Student("유미선", 30)
    );

  }
}
```

```
평균 점수: 20.0
```

스트림 요소 처리

StreamPipeLineExample.java

```
package ch17.sec03;

import java.util.Arrays;
import java.util.List;

public class StreamPipeLineExample {
  public static void main(String[] args) {
    List<Student> list = Arrays.asList(
        new Student("홍길동", 10),
        new Student("신용권", 20),
        new Student("유미선", 30)
        );

    //방법1
    Stream<Student> studentStream = list.stream();
    //중간 처리(학생 객체를 점수로 매핑)
    IntStream scoreStream = studentStream.mapToInt(student -> student.getScore());
    //최종 처리(평균 점수)
    double avg = scoreStream.average().getAsDouble();
    System.out.println("평균 점수: " + avg);
```

```
평균 점수: 20.0
```

스트림 요소 처리

StreamPipeLineExample.java

```
    //방법2
    double avg = list.stream()
        .mapToInt(student -> student.getScore())
        .average()
        .getAsDouble();

    System.out.println("평균 점수: " + avg);
  }
}
```

```
평균 점수: 20.0
```

스트림 요소 처리

- 롬복을 이용하여 전체 매개변수를 가지는 생성자, Getter, Setter, toString을 정의하세요.

```
package ch17.sec04.exam01;

public class Product {
  private int pno;
  private String name;
  private String company;
  private int price;

}
```

Product.java

```
package ch17.sec04.exam01;

@AllArgsConstructor
@Data
public class Product {
  private int pno;
  private String name;
  private String company;
  private int price;

}
```

스트림 요소 처리

스트림을 이용하여 list에 담긴 Product를 출력하세요(toString 이용).

```
package ch17.sec04.exam01;

import java.util.ArrayList;
import java.util.List;

public class StreamExample {
  public static void main(String[] args) {
    //List 컬렉션 생성
    List<Product> list = new ArrayList<>();
    for(int i=1; i<=5; i++) {
      Product product = new Product(i, "상품"+i, "멋진회사", (int)(10000*Math.random()));
      list.add(product);
    }

  }
}
```

스트림 요소 처리

StreamExample.java

```
package ch17.sec04.exam01;

import java.util.ArrayList;
import java.util.List;
import java.util.stream.Stream;

public class StreamExample {
  public static void main(String[] args) {
    //List 컬렉션 생성
    List<Product> list = new ArrayList<>();
    for(int i=1; i<=5; i++) {
      Product product = new Product(i, "상품"+i, "멋진회사", (int)(10000*Math.random()));
      list.add(product);
    }

    //객체 스트림 얻기
    Stream<Product> stream = list.stream();
    stream.forEach(p -> System.out.println(p));
  }
}
```

```
{pno:1, name:상품1, company:멋진회사, price:6188}
{pno:2, name:상품2, company:멋진회사, price:2510}
{pno:3, name:상품3, company:멋진회사, price:9932}
{pno:4, name:상품4, company:멋진회사, price:4317}
{pno:5, name:상품5, company:멋진회사, price:170}
```

스트림 요소 처리

- 아래와 같이 출력되도록 스트림을 이용한 코드를 완성하세요.

```
package ch17.sec04.exam02;

import java.util.Arrays;
import java.util.stream.IntStream;
import java.util.stream.Stream;

public class StreamExample {
  public static void main(String[] args) {
    String[] strArray = { "홍길동", "신용권", "김미나"};



    int[] intArray = { 1, 2, 3, 4, 5 };


  }
}
```

```
홍길동,신용권,김미나,
1,2,3,4,5,
```

스트림 요소 처리

StreamExample.java

```
package ch17.sec04.exam02;

import java.util.Arrays;
import java.util.stream.IntStream;
import java.util.stream.Stream;

public class StreamExample {
  public static void main(String[] args) {
    String[] strArray = { "홍길동", "신용권", "김미나"};
    Stream<String> strStream = Arrays.stream(strArray);
    strStream.forEach(item -> System.out.print(item + ","));
    System.out.println();

    int[] intArray = { 1, 2, 3, 4, 5 };
    IntStream intStream = Arrays.stream(intArray);
    intStream.forEach(item -> System.out.print(item + ","));
    System.out.println();
  }
}
```

```
홍길동,신용권,김미나,
1,2,3,4,5,
```

스트림 요소 처리

1에서 100까지의 합을 구하고, 다음과 같은 출력하세요.

```
package ch17.sec04.exam03;

import java.util.stream.IntStream;

public class StreamExample {
  public static int sum;  // 합계를 구하기 위한 변수

  public static void main(String[] args) {

  }
}
```

```
총합: 5050
```

스트림 요소 처리

StreamExample.java

```
package ch17.sec04.exam03;

import java.util.stream.IntStream;

public class StreamExample {
  public static int sum;

  public static void main(String[] args) {
    IntStream stream = IntStream.rangeClosed(1, 100);
    stream.forEach(a -> sum += a);
    System.out.println("총합: " + sum);
  }
}
```

```
총합: 5050
```

스트림 요소 처리

다음과 같이 출력되도록 코드를 완성하세요.

```
package ch17.sec05;

import java.util.ArrayList;
import java.util.List;

public class FilteringExample {
  public static void main(String[] args) {
    List<String> list = new ArrayList<>();
    list.add("홍길동");
    list.add("신용권");
    list.add("감자바");
    list.add("신용권");
    list.add("신민철");

    //중복 요소 제거

    //신으로 시작하는 요소만 필터링

    //중복 요소를 먼저 제거하고, 신으로 시작하는 요소만 필터링
  }
}
```

```
홍길동
신용권
감자바
신민철

신용권
신용권
신민철

신용권
신민철
```

FilteringExample.java

```
package ch17.sec05;

import java.util.ArrayList;
import java.util.List;

public class FilteringExample {
  public static void main(String[] args) {
    //List 컬렉션 생성
    List<String> list = new ArrayList<>();
    list.add("홍길동");
    list.add("신용권");
    list.add("감자바");
    list.add("신용권");
    list.add("신민철");
```

스트림 요소 처리

FilteringExample.java

```
    //중복 요소 제거
    list.stream()
      .distinct()
      .forEach(n -> System.out.println(n));
    System.out.println();

    //신으로 시작하는 요소만 필터링
    list.stream()
      .filter(n -> n.startsWith("신"))
      .forEach(n -> System.out.println(n));
    System.out.println();

    //중복 요소를 먼저 제거하고, 신으로 시작하는 요소만 필터링
    list.stream()
      .distinct()
      .filter(n -> n.startsWith("신"))
      .forEach(n -> System.out.println(n));
  }
}
```

```
홍길동
신용권
감자바
신민철

신용권
신용권
신민철

신용권
신민철
```

스트림 요소 처리

- score 값으로 크기 비교가 가능한 클래스가 되도록 다음 클래스를 수정하세요..

```
package ch17.sec07.exam01;

public class Student {
  private String name;
  private int score;

  public Student(String name, int score) {
    this.name = name;
    this.score = score;
  }

  public String getName() { return name; }

  public int getScore() { return score; }

}
```

Student.java

```
package ch17.sec07.exam01;

public class Student implements Comparable<Student> {
  private String name;
  private int score;

  public Student(String name, int score) {
    this.name = name;
    this.score = score;
  }

  public String getName() { return name; }
  public int getScore() { return score; }

  @Override
  public int compareTo(Student o) {
    return Integer.compare(score, o.score);
  }
}
```

스트림 요소 처리

Student 리스트를 성적순으로 오름차순, 내림차순으로 각각 정렬한 후 출력하세요.

```
package ch17.sec07.exam01;

import java.util.ArrayList;
import java.util.Comparator;
import java.util.List;

public class SortingExample {
  public static void main(String[] args) {
    List<Student> studentList = new ArrayList<>();
    studentList.add(new Student("홍길동", 30));
    studentList.add(new Student("신용권", 10));
    studentList.add(new Student("유미선", 20));

    //점수를 기준으로 오름차순으로 정렬한 새 스트림 얻기

    //점수를 기준으로 내림차순으로 정렬한 새 스트림 얻기

  }
}
```

```
신용권: 10
유미선: 20
홍길동: 30

홍길동: 30
유미선: 20
신용권: 10
```

스트림 요소 처리

SortingExample.java

```
package ch17.sec07.exam01;

import java.util.ArrayList;
import java.util.Comparator;
import java.util.List;

public class SortingExample {
  public static void main(String[] args) {
    //List 컬렉션 생성
    List<Student> studentList = new ArrayList<>();
    studentList.add(new Student("홍길동", 30));
    studentList.add(new Student("신용권", 10));
    studentList.add(new Student("유미선", 20));

    //점수를 기준으로 오름차순으로 정렬한 새 스트림 얻기
    studentList.stream()
        .sorted( )
        .forEach(s -> System.out.println(s.getName() + ": " + s.getScore()));
    System.out.println();
```

```
신용권: 10
유미선: 20
홍길동: 30
```

스트림 요소 처리

다음과 같이 클래스를 정의하세요.

```
package ch17.sec07.exam02;

public class Student {
  private String name;
  private int score;

  public Student(String name, int score) {
    this.name = name;
    this.score = score;
  }

  public String getName() { return name; }
  public int getScore() { return score; }
}
```

스트림 요소 처리

- **Comparator 인터페이스를 이용하여 Student 리스트를 성적순으로 오름차순, 내림차순으로 각각 정렬한 후 출력하세요.**

```
package ch17.sec07.exam01;

import java.util.ArrayList;
import java.util.Comparator;
import java.util.List;

public class SortingExample {
  public static void main(String[] args) {
    List<Student> studentList = new ArrayList<>();
    studentList.add(new Student("홍길동", 30));
    studentList.add(new Student("신용권", 10));
    studentList.add(new Student("유미선", 20));

    //점수를 기준으로 오름차순으로 정렬한 새 스트림 얻기

    //점수를 기준으로 내림차순으로 정렬한 새 스트림 얻기

  }
}
```

```
신용권: 10
유미선: 20
홍길동: 30

홍길동: 30
유미선: 20
신용권: 10
```

SortingExample.java

```
package ch17.sec07.exam02;
…
public class SortingExample {
  public static void main(String[] args) {
    //List 컬렉션 생성
    List<Student> studentList = new ArrayList<>();
    studentList.add(new Student("홍길동", 30));
    studentList.add(new Student("신용권", 10));
    studentList.add(new Student("유미선", 20));

    //점수를 기준으로 오름차순으로 정렬한 새 스트림 얻기
    studentList.stream()
      .sorted((s1, s2) -> Integer.compare(s1.getScore(), s2.getScore()))
      .forEach(s -> System.out.println(s.getName() + ": " + s.getScore()));
    System.out.println();

    //점수를 기준으로 내림차순으로 정렬한 새 스트림 얻기
    studentList.stream()
      .sorted((s1, s2) -> Integer.compare(s2.getScore(), s1.getScore()))
      .forEach(s -> System.out.println(s.getName() + ": " + s.getScore()));
  }
}
```

```
신용권: 10
유미선: 20
홍길동: 30

홍길동: 30
유미선: 20
신용권: 10
```

스트림 요소 처리

다음과 같이 출력되도록 코드를 작성하세요.

```
package ch17.sec10;

import java.util.Arrays;

public class AggregateExample {
  public static void main(String[] args) {
    int[] arr = {1, 2, 3, 4, 5};

    //2의 배수 카운팅

    //2의 배수 총합

    //2의 배수 평균

    //2의 배수 중 최대값

    //2의 배수 중 최소값

    //첫 번째 3의 배수

  }
}
```

```
2의 배수 개수: 2
2의 배수의 합: 6
2의 배수의 평균: 3.0
최대값: 4
최소값: 2
첫 번째 3의 배수: 3
```

스트림 요소 처리

AggregateExample.java

```
package ch17.sec10;

import java.util.Arrays;

public class AggregateExample {
  public static void main(String[] args) {
    //정수 배열
    int[] arr = {1, 2, 3, 4, 5};

    //카운팅
    long count = Arrays.stream(arr)
        .filter(n -> n%2==0)
        .count();
    System.out.println("2의 배수 개수: " + count);

    //총합
    long sum = Arrays.stream(arr)
        .filter(n -> n%2==0)
        .sum();
    System.out.println("2의 배수의 합: " + sum);
```

```
2의 배수 개수: 2
2의 배수의 합: 6
```

AggregateExample.java

```
//평균
double avg = Arrays.stream(arr)
    .filter(n -> n%2==0)
    .average()
    .getAsDouble();
System.out.println("2의 배수의 평균: " + avg);

//최대값
int max = Arrays.stream(arr)
    .filter(n -> n%2==0)
    .max()
    .getAsInt();
System.out.println("최대값: " + max);

//최소값
int min = Arrays.stream(arr)
    .filter(n -> n%2==0)
    .min()
    .getAsInt();
System.out.println("최소값: " + min);
```

```
2의 배수의 평균: 3.0
최대값: 4
최소값: 2
```

스트림 요소 처리

AggregateExample.java

```
    //첫 번째 요소
    int first = Arrays.stream(arr)
        .filter(n -> n%3==0)
        .findFirst()
        .getAsInt();
    System.out.println("첫 번째 3의 배수: " + first);
  }
}
```

```
첫 번째 3의 배수: 3
```

스트림 요소 처리

- **다음 코드에서 리스트에 담긴 정수의 평균을 구하여 출력하세요.**
  - 리스트의 요소는 수정하지 않음
  - null 때문에 평균을 구하지 못하는 경우 디폴트 값으로 0.0을 설정함

```
package ch17.sec10;

import java.util.ArrayList;
import java.util.List;
import java.util.OptionalDouble;

public class OptionalExample {
  public static void main(String[] args) {
    List<Integer> list = new ArrayList< >();

    double avg;


    System.out.println("평균: " + avg);
  }
}
```

```
평균: 0.0
```

OptionalExample.java

```
package ch17.sec10;

import java.util.ArrayList;
import java.util.List;
import java.util.OptionalDouble;

public class OptionalExample {
  public static void main(String[] args) {
    List<Integer> list = new ArrayList< >();

    /*//예외 발생(java.util.NoSuchElementException)
    double avg = list.stream()
      .mapToInt(Integer :: intValue)
      .average()
      .getAsDouble();
    */
```

OptionalExample.java

```
    //방법1
    OptionalDouble optional = list.stream()
      .mapToInt(Integer :: intValue)
      .average();
    if(optional.isPresent()) {
      System.out.println("방법1_평균: " + optional.getAsDouble());
    } else {
      System.out.println("방법1_평균: 0.0");
    }

    //방법2
    double avg = list.stream()
      .mapToInt(Integer :: intValue)
      .average()
      .orElse(0.0);
    System.out.println("방법2_평균: " + avg);

    //방법3
    list.stream()
      .mapToInt(Integer :: intValue)
      .average()
      .ifPresent(a -> System.out.println("방법3_평균: " + a));
  }
}
```

```
방법1_평균: 0.0
방법2_평균: 0.0
```