자바프로그래밍및실습 과제 3

214823 박종현

June 2022

```java
/**
 * @author @ShapeLayer
 */

/*
  Average.java
  CLI 환경에서 실행하고자 한다면 `java Average.java 1 2 3 5 7 8 1 2 3 4`를
↪  입력하세요.
*/

public class Main {
  public static void main(String[] args) {
    int sums = 0; // 합계 기록할 변수 선언
    // `args` 변수의 길이만큼 반복문 시작
    for (int i = 0; i < args.length; i++) {
      // `args[i]`는 String형 값이므로 int형으로 캐스팅한 뒤 `sums`에 가산
      sums += Integer.parseInt(args[i]);
    }
    // `sums`와 `args.length` 모두 정수이므로 둘 중 하나를 실수로 캐스팅해야 실수
↪  계산이 수행됨
    // 실수로 캐스팅하지않으면 몫 연산 수행
    System.out.println("입력받은 인자 값의 평균은 : " + sums/(float)args.length);
  }
}

/**
 * @author @ShapeLayer
 */

import java.util.Scanner;

public class Main {
  public static void main(String[] args) {
    // 표준 입력을 위해 Scanner 객체 선언
    Scanner sc = new Scanner(System.in);

    // 문제 제시 값
    String course[] = {"java", "c++", "HTML5", "컴퓨터구조", "안드로이드"};
    int score[] = {95, 88, 76, 62, 55};
    String name;

    // 입력값이 "그만"일 때까지 프로그램이 작동해야하므로 무한 루프 시작
    while(true) {
      // 과목 이름 입력
      System.out.print("과목 이름 >> ");
      name = sc.next();

      // 입력값이 "그만"인 경우 프로그램 종료
      if (name.equals("그만")) { break; }

      // 검색 중 입력값에 해당하는 값이 존재하고 출력했는지 여부를 확인하기 위해
      // 부울형 `flag` 변수 선언
      // 아직 존재하는지 확인하지 않았고, 결과를 출력하지 않았으므로 false로 지정
      boolean flag = false;
      for (int i = 0; i < course.length; i++) {
        // 과제 첨부 파일의 "실행 예"에서 Java를 입력했는데 java의 결과가 출력되는
↪  것을 확인할 수 있음
        // 따라서 이 프로그램은 대소문자를 구분하지 않는 것으로 처리함
        // * 입력값과 키(과목)값 모두 소문자로 변환한 후 두 값이 같은지 확인
        if (course[i].toLowerCase().equals(name.toLowerCase())) {
          System.out.println(String.format("%s의 점수는 %d", name, score[i]));
```

```
          flag = true; // 대상이 존재하므로 `flag`를 true로 변경
        }
      }
      // `flag`가 false라면 대상이 존재하지 않고 결과를 출력한 사실도 없으므로 없는
↪   과목이라고 출력
      if (!flag) System.out.println("없는 과목입니다.");
    }
  }
}

/**
 * @author @ShapeLayer
 */

import java.util.Scanner;
// 예외 처리를 위해 InputMismatchException 클래스 임포트
import java.util.InputMismatchException;

public class Main {
  public static void main(String[] args) {
    int n, m;

    // 문제 제시 코드
    Scanner scanner = new Scanner(System.in);
    while (true) {
      System.out.print("곱하고자 하는 두수 입력>>");

      // 예외 처리를 위해 try-catch문 사용
      try {
        n = scanner.nextInt();
        m = scanner.nextInt();
        break;
      } catch (InputMismatchException e) {
        // `InputMismatchException` 오류를 캐치하도록 지정.
        // e는 오류(error) 정보를 담고 있음
        System.out.println("실수는 입력하면 안됩니다.");
        scanner.nextLine();
      }
    }
    System.out.println(n + "x" + m + "=" + n*m);
    scanner.close();
  }
}

/**
 * @author @ShapeLayer
 */

import java.util.Scanner;

public class Main {
  public static void main(String[] args) {
    // 표준 입력을 위해 Scanner 객체 선언
    Scanner sc = new Scanner(System.in);
    String gets;

    // 문제 제시 값
    String str[] = {"가위", "바위", "보"};

    System.out.println("컴퓨터와 가위 바위 보 게임을 합니다.");
    // "그만"이 입력될때까지 반복해야하므로 무한 루프 선언
    while (true) {
```

```
      System.out.print("가위 바위 보! >> ");
      gets = sc.next();
      // 문제 제시 코드
      int n = (int)(Math.random()*3);

      // "그만"이 입력된다면 무한 루프 종료
      if (gets.equals("그만")) break;

      // 문제에서 gets.equals()를 사용하여 값을 확인하라고 지정하였으니 아래와 같이
↪   케이스별로 처리
      if (str[n].equals("가위")) {
        if (gets.equals("가위"))
          System.out.println(String.format("사용자 = %s, 컴퓨터 %s, 비겼습니다.",
↪   gets, str[n]));
        if (gets.equals("바위"))
          System.out.println(String.format("사용자 = %s, 컴퓨터 %s, 사용자가
↪   이겼습니다.", gets, str[n]));
        if (gets.equals("보"))
          System.out.println(String.format("사용자 = %s, 컴퓨터 %s, 컴퓨터가
↪   이겼습니다.", gets, str[n]));
      }
      else if (str[n].equals("바위")) {
        if (gets.equals("가위"))
          System.out.println(String.format("사용자 = %s, 컴퓨터 %s, 컴퓨터가
↪   이겼습니다.", gets, str[n]));
        if (gets.equals("바위"))
          System.out.println(String.format("사용자 = %s, 컴퓨터 %s, 비겼습니다.",
↪   gets, str[n]));
        if (gets.equals("보"))
          System.out.println(String.format("사용자 = %s, 컴퓨터 %s, 사용자가
↪   이겼습니다.", gets, str[n]));
      }
      else if (str[n].equals("보")) {
        if (gets.equals("가위"))
          System.out.println(String.format("사용자 = %s, 컴퓨터 %s, 사용자가
↪   이겼습니다.", gets, str[n]));
        if (gets.equals("바위"))
          System.out.println(String.format("사용자 = %s, 컴퓨터 %s, 컴퓨터가
↪   이겼습니다.", gets, str[n]));
        if (gets.equals("보"))
          System.out.println(String.format("사용자 = %s, 컴퓨터 %s, 비겼습니다.",
↪   gets, str[n]));
      }
      // fallback
      //  "가위", "바위", "보" 외의 값을 입력했을 때 처리는 제시하지 않았으므로
      //  별도의 처리를 거치지 않고 루프를 계속함 (= 계속해서 입력을 받아옴)
    }
    System.out.println("게임을 종료합니다.");
  }
}

/**
 * @author @ShapeLayer
 */

// 코드 Javadoc 문서화
// 참조:
↪  https://docs.oracle.com/en/java/javase/17/docs/specs/javadoc/doc-comment-spec.html

import java.util.Scanner;
```

```java
public class Main {
  public static void main(String[] args) {
    // 문제 제시 코드
    TV myTV = new TV("LG", 2017, 32);
    myTV.show();
  }
}

/**
 * TV 클래스
 *
 * TV 클래스는 생산된 TV에 대한 정보를 담을 수 있습니다.
 * 이름, 생산자(생산업체), 인치 정보를 담고 처리할 수 있고, 그 내용을 출력할 수
↪  있습니다.
 *
 * @author 박종현
 * @since JDK-8
 */
class TV {
  /**
   * TV 이름을 표현합니다.
   */
  String name;

  /**
   * 생산자(생산업체)를 표현합니다.
   */
  int manufactures;

  /**
   * 크기를 인치 단위로 표현합니다.
   */
  int inch;

  /**
   * TV 객체를 생성합니다.
   *
   * @param name TV의 이름
   * @param manufactures 생산자(생산업체)
   * @param inch 크기(인치)
   */
  public TV(String name, int manufactures, int inch) {
    // `this.name`은 TV 객체의 name 속성, `name`은 생성자의 매개변수
    // 아래와 같이 작성한다면 생성자의 매개변수를 객체의 각 속성으로 저장할 수 있음
    this.name = name;
    this.manufactures = manufactures;
    this.inch = inch;
  }

  /**
   * TV 정보를 콘솔에 표준 출력합니다.
   *
   * @return {@code null}
   */
  public void show() {
    System.out.println(String.format("%s에서 만든 %d년형 %d인치 TV", this.name,
↪  this.manufactures, this.inch));
  }
}

/**
```

```java
 * @author @ShapeLayer
 */

// 코드 Javadoc 문서화
// 참조:
↪  https://docs.oracle.com/en/java/javase/17/docs/specs/javadoc/doc-comment-spec.html

import java.util.Scanner;

public class Main {
  public static void main(String[] args) {
    // 문제 제시 코드
    Scanner scanner = new Scanner(System.in);
    System.out.print("수학, 과학, 영어순으로 3개의 점수 입력>>");
    int math = scanner.nextInt();
    int science = scanner.nextInt();
    int english = scanner.nextInt();
    Grade me = new Grade(math, science, english);
    System.out.println("평균은 " + me.average());
    // average()는 평균을 계산하여 리턴
    scanner.close();
  }
}

/**
 * Grade 클래스
 * Grade 클래스는 수학, 과학, 영어 세 과목의 점수를 처리할 수 있는 클래스입니다.
 * 세 과목을 한 번에 관리할 수 있고, 세 과목의 평균을 쉽게 확인할 수 있도록
↪  지원합니다.
 * @author 박종현
 */
class Grade {
  // 자바에서 접근 제어자는 기본값이 private이므로 접근 제어자를 생략하면 기본값
↪  private로 지정됨

  /**
   * 수학 과목 점수
   */
  int math;

  /**
   * 과학 과목 점수
   */
  int science;

  /**
   * 영어 과목 점수
   */
  int english;

  /**
   * Grade 객체를 생성합니다.
   * @param math 수학 과목 점수
   * @param science 과학 과목 점수
   * @param english 영여 과목 점수
   */
  public Grade(int math, int science, int english) {
    this.math = math;
    this.science = science;
    this.english = english;
  }
```

```java
  /**
   * 수학 과학 영어 세 과목의 평균값을 정수로 반환합니다.
   * @return {@code int} 세 과목의 평균값
   */
  int average() {
    return (this.math + this.science + this.english) / 3;
  }
}

/**
 * @author @ShapeLayer
 */

// 코드 Javadoc 문서화
// 참조:
↪  https://docs.oracle.com/en/java/javase/17/docs/specs/javadoc/doc-comment-spec.html

import java.util.Scanner;

/**
 * Phonebook 클래스
 *
 * Phonebook 클래스는 주소록 애플리케이션 클래스입니다.
 * 사용자는 다른 사람들의 프로필을 등록하고, 등록한 프로필을 참조할 수 있습니다.
 */
public class Phonebook {
  /**
   * 프로그램 진입점
   * @param args
   */
  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.print("인원수 >> ");

    // 인원수 입력 받아옴
    int n = sc.nextInt();
    // 입력값을 크기로 갖는 Phone 배열 생성
    Phone[] phones = new Phone[n];

    /* 프로필(전화번호부) 등록 프로세스 */
    // for문으로 배열의 요소들을 순차적으로 탐색
    // 사용자로부터 입력받고, 입력받은 값으로 초기화 진행
    for (int i = 0; i < n; i++) {
      System.out.print("이름과 전화번호 (이름과 번호는 빈 칸없이 입력 >>> ");
      String name = sc.next();
      String tel = sc.next();
      // Phone 배열의 i번째 요소를 사용자의 입력값으로 초기화
      phones[i] = new Phone(name, tel);
    }
    System.out.println("저장되었습니다...");

    // "그만"이 입력될 때까지 계속해서 검색해야하므로 무한 루프 시작
    while (true) {
      System.out.print("검색할 이름 >> ");
      String gets = sc.next();

      // 입력값이 "그만"이라면 루프 종료
      if (gets.equals("그만")) break;

      // 검색 결과를 잘 찾아내고 출력했는지 확인하기위해 `flag` 변수 선언
```

```
      // 선언 시기에는 아직 검색을 시작하지 않았으므로 false로 설정
      boolean flag = false;

      // 검색 시작
      for (int i = 0; i < n; i++) {
        if (phones[i].getName().equals(gets)) {
          // 검색 결과가 존재하므로 `flag`를 true로 지정
          flag = true;
          // 결과 출력
          System.out.println(String.format("%s의 번호는 %s입니다.",
↪   phones[i].getName(), phones[i].getTel()));
        }
      }
      // `flag`가 `false`라면 검색 결과가 존재하지 않았다는 것이므로 fallback 처리
      if (!flag) { System.out.println(String.format("%s 이(가) 없습니다.",
↪   gets)); }
    }
    System.out.println("프로그램을 종료합니다.");
  }
}

/**
 * Phone 클래스
 *
 * Phone 클래스는 Phonebook 클래스가 관리하는 프로필 단위입니다.
 * 하나의 프로필은 하나의 Phone 객체입니다.
 */
class Phone {
  /**
   * 대상의 이름입니다.
   */
  private String name;
  /**
   * 대상의 전화번호 정보입니다.
   */
  private String tel;

  /**
   * Phone 객체를 생성합니다.
   * @param name 대상의 이름
   * @param tel 대상의 전화번호
   */
  public Phone(String name, String tel) {
    this.name = name; this.tel = tel;
  }

  /**
   * {@code getter} 이름을 반환합니다.
   * @return {@code String} 이름
   */
  public String getName() { return name; }

  /**
   * {@code getter} 전화번호를 반환합니다.
   * @return {@code String} 전화번호
   */
  public String getTel() { return tel; }
}

/**
 * @author @ShapeLayer
```

```java
 */

// 코드 Javadoc 문서화
// 참조:
↪  https://docs.oracle.com/en/java/javase/17/docs/specs/javadoc/doc-comment-spec.html

import java.util.Arrays;

public class StaticEx {
  public static void main(String[] args) {
    // 문제 제시 코드
    int[] array1 = { 1, 5, 7, 9 };
    int[] array2 = { 3, 6, -1, 100, 77 };
    int[] array3 = ArrayUtil.concat(array1, array2);
    ArrayUtil.print(array3);
  }
}

/**
 * ArrayUtil 클래스
 *
 * ArrayUtil 클래스는 배열을 좀 더 쉽게 제어할 수 있는 메서드를 포함하고 있습니다.
 */
class ArrayUtil {
  /**
   * 두 정수 배열을 하나로 병합합니다.
   * @param a {@code int[]} 첫 번째 정수 배열
   * @param b {@code int[]} 두 번째 정수 배열
   * @return {@code int[]} 배열 병합 결과
   */
  public static int[] concat(int[] a, int[] b) {
    // 배열 a와 b를 연결한 새로운 배열 리턴
    int[] newArr = new int[a.length + b.length];
    System.arraycopy(a, 0, newArr, 0, a.length);
    System.arraycopy(b, 0, newArr, a.length, b.length);

    /* 시간복잡도 O(n)인 처리 방법
    for (int i = 0; i < a.length; i++) newArr[i] = a[i];
    for (int i = 0; i < b.length; i++) newArr[a.length + i] = b[i];
    */
    return newArr;
  }

  /**
   * 정수 배열을 출력합니다.
   *
   * @param a {@code int[]} 정수 배열
   */
  public static void print(int[] a) {
    // 배열 a를 출력
    // System.out.println(Arrays.toString(a));

    // print()는 출력 후 개행하지 않으므로 한 줄을 여러 줄에 걸쳐서 출력
    System.out.print("[ ");
    for (int i = 0; i < a.length; i++)
      System.out.print(a[i] + " ");
    System.out.println("]");
  }
}

/**
```

```java
 * @author @ShapeLayer
 */

// 코드 Javadoc 문서화
// 참조:
↪  https://docs.oracle.com/en/java/javase/17/docs/specs/javadoc/doc-comment-spec.html

import java.util.Scanner;

/**
 * DicApp 클래스
 *
 * DicApp은 사전 애플리케이션 클래스입니다.
 * 사용자는 사전을 로드하여 사전에 기록되어 있는 단어를 검색할 수 있습니다.
 */
public class DicApp {
  /**
   * 단어가 한글 자모로 구성될 때 마지막 글자에 종성이 존재하는지 확인합니다.
   *
   * @param word 확인할 단어
   * @return 마지막 글자에 종성이 존재하는지 여부
   */
  public static boolean isComplete(String word) {
    char lastWord = word.charAt(word.length() - 1);
    // 마지막 글자가 한국어가 아닌 경우 `false` 리턴
    if (lastWord < 0xAC00 || lastWord > 0xD7A3) { return false; }
    // 한글 자모를 분리하여 종성이 존재하는지 확인
    // 참조 https://hanpsy.tistory.com/2
    return (lastWord - 0xAC00) % 28 > 0; // 0인 경우 = 종성 데이터 없음 (= 종성
↪  없음)
  }

  /**
   * 프로그램 진입점
   * @param args
   */
  public static void main(String[] args) {
    System.out.println("한영 단어 검색 프로그램입니다.");
    Scanner sc = new Scanner(System.in);

    // "종료"가 입력될 때까지 프로그램이 작동해야하므로 무한 루프 시작
    while (true) {
      System.out.print("한글 단어? ");
      String gets = sc.next();
      // "종료" 입력시 무한 루프 종료
      if (gets.equals("종료")) break;

      // Dictionary.kor2Eng 메서드는 정적 메서드이므로 객체를 생성하지 않고 호출
      // 결과가 있다면 String형으로 반환, 없다면 null 반환
      String res = Dictionary.kor2Eng(gets);
      System.out.println(String.format("%s%s %s",
        gets,
        DicApp.isComplete(gets) ? "은" : "는", // 은/는 여부 확인 메서드 호출.
↪  종성이 있다면 "은", 없다면 "는"
        res != null ? res : "저의 사전에 없습니다." // kor2Eng 결과가 null이
↪  아니라면 그대로 출력, null이라면 결과 없음 출력
      ));
    }
    System.out.println("프로그램을 종료합니다.");
  }
}
```

```java
/**
 * Dictionary 클래스
 *
 * Dictionary 클래스는 특정 문구를 키 값 삼아 대응되는 다른 문구로 변경할 수 있는
↪  메서드를 제공합니다.
 */
class Dictionary {
  // 문제 제시 코드
  /**
   * 한국어 단어를 기록합니다.
   */
  private static String[] kor = {"사랑", "아기", "돈", "미래", "희망"};
  /**
   * 영어 단어를 기록합니다.
   */
  private static String[] eng = {"love", "baby", "money", "future", "hope"};

  /**
   * 입력받은 한국어 단어를 영어 단어로 반환합니다.
   * 만약 목록에 등록되어있지 않은 단어라면 {@code null}을 반환합니다.
   *
   * @param word
   * @return {@code String?} 영단어 조회 결과를 반환합니다.
   */
  public static String kor2Eng(String word) {
    // 시간 복잡도 O(n)의 처리 알고리즘 구현
    for (int i = 0; i < kor.length; i++) {
      // 만약 `word`(메서드의 매개변수)가 배열에 있다면 변환 결과 리턴
      if (kor[i].equals(word)) {
        return eng[i];
      }
    }

    // 이 부분까지 도달했다면 단어 목록에 대상 단어가 존재하지 않음을 의미함
    // 단어가 존재하지 않으므로 null 반환
    return null;
  }
}

/**
 * @author @ShapeLayer
 */

// 코드 Javadoc 문서화
// 참조:
↪  https://docs.oracle.com/en/java/javase/17/docs/specs/javadoc/doc-comment-spec.html

public class Main {
  public static void main(String [] args) {
    // ColorTV 객체 생성
    ColorTV myTV = new ColorTV(32, 1024);
    myTV.printProperty();
  }
}

/**
 * TV 클래스
 *
 * TV 클래스는 생산된 TV에 대한 정보를 담습니다.
 */
```

```java
class TV {
  /**
   * TV의 크기(인치)입니다.
   */
  private int size;

  /**
   * TV 객체를 생성합니다.
   * @param size 크기(인치)
   */
  public TV (int size) { this.size = size; }

  /**
   * TV의 크기를 반환합니다.
   * @return {@code int} TV의 크기(인치)
   */
  protected int getSize() { return size; }
}

/**
 * ColorTV 클래스
 *
 * TV 클래스를 상속함
 * ColorTV 클래스는 생산된 컬러TV에 대한 정보를 담습니다.
 */
class ColorTV extends TV {
  /**
   * TV가 표현할 수 있는 색상의 정보입니다.
   */
  private int colors;

  /**
   * ColorTV 객체를 생성합니다.
   * @param size {@code int} 크기(인치)
   * @param colors {@code int} 표현 가능한 색상 수
   */
  public ColorTV (int size, int colors) {
    super(size);
    this.colors = colors;
  }

  /**
   * TV가 표현 가능한 색상의 개수를 반환합니다.
   * @return {@code int} 표현 가능한 색상 수
   */
  protected int getColors() {
    return this.colors;
  }

  /**
   * TV의 속성을 표준 출력합니다.
   */
  public void printProperty() {
    System.out.println(String.format("%d인치 %d컬러", this.getSize(),
↪   this.getColors()));
  }
}

/**
 * @author @ShapeLayer
 */
```

```java
// 코드 Javadoc 문서화
// 참조:
↪  https://docs.oracle.com/en/java/javase/17/docs/specs/javadoc/doc-comment-spec.html

public class Main {
  public static void main(String [] args) {
    IPTV iptv = new IPTV("192.1.1.2", 32, 2048);
    iptv.printProperty();
  }
}

/**
 * TV 클래스
 *
 * TV 클래스는 생산된 TV에 대한 정보를 담습니다.
 */
class TV {
  /**
   * TV의 크기(인치)입니다.
   */
  private int size;

  /**
   * TV 객체를 생성합니다.
   * @param size 크기(인치)
   */
  public TV (int size) { this.size = size; }

  /**
   * TV의 크기를 반환합니다.
   * @return {@code int} TV의 크기(인치)
   */
  protected int getSize() { return size; }
}

/**
 * ColorTV 클래스
 *
 * TV 클래스를 상속함
 * ColorTV 클래스는 생산된 컬러TV에 대한 정보를 담습니다.
 */
class ColorTV extends TV {
  /**
   * TV가 표현할 수 있는 색상의 정보입니다.
   */
  private int colors;

  /**
   * ColorTV 객체를 생성합니다.
   * @param size {@code int} 크기(인치)
   * @param colors {@code int} 표현 가능한 색상 수
   */
  public ColorTV (int size, int colors) {
    super(size);
    this.colors = colors;
  }

  /**
   * TV가 표현 가능한 색상의 개수를 반환합니다.
   * @return {@code int} 표현 가능한 색상 수
```

```java
    */
    protected int getColors() {
        return this.colors;
    }

    /**
     * TV의 속성을 표준 출력합니다.
     */
    public void printProperty() {
        System.out.println(String.format("%d인치 %d컬러", this.getSize(),
↪       this.getColors()));
    }
}

/**
 * IPTV 클래스
 *
 * ColorTV를 상속함
 * IPTV 클래스는 생산된 IPTV에 대한 정보를 담습니다.
 */
class IPTV extends ColorTV {
    /**
     * TV의 IP주소
     */
    private String ip;

    /**
     * IPTV 객체를 생성합니다.
     * @param ip {@code String} IP주소
     * @param size {@code int} 크기(인치)
     * @param colors {@code int} 표현 가능한 색상 수
     */
    IPTV(String ip, int size, int colors) {
        super(size, colors);
        this.ip = ip;
    }

    /**
     * TV에 할당된 IP 주소를 반환합니다.
     * @return {@code String} IP 주소
     */
    protected String getIP() { return this.ip; }

    /**
     * TV의 속성을 표준 출력합니다.
     */
    public void printProperty() {
        System.out.println("나의 IPTV는 "+this.getIP()+" 주소의 "+this.getSize()+"
↪       인치 "+this.getColors()+"컬러");
    }
}

/**
 * @author @ShapeLayer
 */

// 코드 Javadoc 문서화
// 참조:
↪   https://docs.oracle.com/en/java/javase/17/docs/specs/javadoc/doc-comment-spec.html

public class Main {
```

```
  public static void main(String[] args) {
    // 문제 제시 코드
    ColorPoint zeroPoint = new ColorPoint(); // (0,0) 위치의 BLACK 색점
    System.out.println(zeroPoint.toString() + "입니다.");
    ColorPoint cp = new ColorPoint(10, 10); // (10,10) 위치의 BLACK 색점
    cp.setXY(5, 5);
    cp.setColor("RED");
    System.out.println(cp.toString() + "입니다.");
  }
}

/**
 * Point 클래스
 *
 * Point 클래스는 2차원 상의 한 점을 표현하는데 활용되는 클래스입니다.
 */
class Point {
  /**
   * 점의 x, y좌표입니다.
   */
  private int x, y;

  /**
   * Point 객체를 생성합니다.
   * @param x x좌표
   * @param y y좌표
   */
  public Point(int x, int y) { this.x = x; this.y = y; }

  /**
   * x좌표를 반환합니다.
   * @return {@code int} 점의 x좌표
   */
  public int getX() { return x; }

  /**
   * y좌표를 반환합니다.
   * @return {@code int} 점의 y좌표
   */
  public int getY() { return y; }

  /**
   * 점의 x좌표와 y좌표를 이동시킵니다. (= setter)
   * @param x 새 x좌표
   * @param y 새 y좌표
   */
  protected void move(int x, int y) { this.x = x; this.y = y; }
}

/**
 * ColorPoint 클래스
 *
 * Point 클래스를 상속함
 * ColorPoint 클래스는 색상을 갖고 있는 한 점을 표현하는데 활용되는 클래스입니다.
 */
class ColorPoint extends Point {
  /**
   * 점의 색상, 기본값은 {@code "BLACK"}입니다.
   */
  private String color = "BLACK";
```

```java
  /**
   * ColorPoint 객체를 생성합니다.
   * 생성된 객체의 x, y좌표는 (0, 0)입니다.
   */
  public ColorPoint() { this(0, 0); }
  /**
   * ColorPoint 객체를 생성합니다.
   * @param x x좌표
   * @param y y좌표
   */
  public ColorPoint(int x, int y) { super(x, y); }

  /**
   * 객체의 x좌표와 y좌표를 설정합니다. (= setter)
   * @param x x좌표
   * @param y y좌표
   */
  public void setXY(int x, int y) { this.move(x, y); }

  /**
   * 객체의 색상을 설정합니다. (= setter)
   * @param color 색상
   */
  public void setColor(String color) { this.color = color; }

  /**
   * 객체의 정보를 String형으로 변환해 반환합니다.
   * @return {@code String} 객체의 정보
   */
  public String toString() { return String.format("%s색의 (%d, %d)의 점",
↪   this.color, this.getX(), this.getY()); }
}

/**
 * @author @ShapeLayer
 */

// 코드 Javadoc 문서화
// 참조:
↪   https://docs.oracle.com/en/java/javase/17/docs/specs/javadoc/doc-comment-spec.html

public class Main {
  public static void main(String[] args) {
    Point3D p = new Point3D(1,2,3); // 1,2,3은 각각 x, y, z 축의 값.
    System.out.println(p.toString() + "입니다.");
    p.moveUp(); // z축 1값 증가
    System.out.println(p.toString() + "입니다.");
    p.moveDown(); // z축 1값 감소
    p.move(10, 10); // x, y 축의 해당 위치로 이동
    System.out.println(p.toString() + "입니다.");
    p.move(100, 200, 300); // x, y, z 축의 해당 위치로 이동
    System.out.println(p.toString() + "입니다.");
  }
}

/**
 * Point 클래스
 *
 * Point 클래스는 2차원 상의 한 점을 표현하는데 활용되는 클래스입니다.
 */
class Point {
```

```
  /**
   * 점의 x, y좌표입니다.
   */
  private int x, y;

  /**
   * Point 객체를 생성합니다.
   * @param x x좌표
   * @param y y좌표
   */
  public Point(int x, int y) { this.x = x; this.y = y; }

  /**
   * x좌표를 반환합니다.
   * @return {@code int} 점의 x좌표
   */
  public int getX() { return x; }

  /**
   * y좌표를 반환합니다.
   * @return {@code int} 점의 y좌표
   */
  public int getY() { return y; }

  /**
   * 점의 x좌표와 y좌표를 이동시킵니다. (= setter)
   * @param x {@code 새 x좌표}
   * @param y {@code 새 y좌표}
   */
  protected void move(int x, int y) { this.x = x; this.y = y; }
}

/**
 * Point3D 클래스
 *
 * Point 클래스를 상속함
 * Point3D 클래스는 3차원 상의 한 점을 표현하는데 활용되는 클래스입니다.
 */
class Point3D extends Point {
  /**
   * 점의 z 좌표입니다.
   */
  private int z;

  /**
   * Point3D 객체를 생성합니다.
   * @param x x좌표
   * @param y y좌표
   * @param z z좌표
   */
  public Point3D(int x, int y, int z) {
    // 부모의 생성자 호출
    super(x, y);
    this.z = z;
  }

  /**
   * 점의 x좌표, y좌표, z좌표를 이동시킵니다. (= setter)
   * @param x 새 x좌표
   * @param y 새 y좌표
   * @param z 새 z좌표
```

```
     */
    public void move(int x, int y, int z) { super.move(x, y); setZ(z); }

    /**
     * z좌표값을 1 증가시킵니다.
     */
    public void moveUp() { this.setZ(this.getZ()+1); }
    /**
     * z좌표값을 1 감소시킵니다.
     */
    public void moveDown() { this.setZ(this.getZ()-1); }
    /**
     * z좌표를 반환합니다.
     * @return {@code int} 점의 z좌표
     */
    public int getZ() { return this.z; }
    /**
     * {@code setter} z좌표를 설정합니다.
     * @param z {@code int} 새 z좌표
     */
    public void setZ(int z) { this.z = z; }

    /**
     * 객체의 정보를 String형으로 변환해 반환합니다.
     * @return {@code String} 객체의 정보
     */
    public String toString() { return String.format("(%d, %d, %d)의 점",
↪    this.getX(), this.getY(), this.getZ()); }
}
```