열거형 – Enum

문자열과 타입 안전성1

- 등급 할인 요구사항
  - BASIC 10% 할인
  - GOLD 20% 할인
  - DIAMOND 30% 할인

예) GOLD 유저가 10000원을 구매하면 할인 대상 금액은 2000원이다.

```
public class DiscountService {
    public int discount(String grade, int price) {
        int discountPercent = 0;

        if (grade.equals("BASIC")) {           discountPercent = 10;
        } else if (grade.equals("GOLD")) {     discountPercent = 20;
        } else if (grade.equals("DIAMOND")) {  discountPercent = 30;
        } else {                               System.out.println(grade + ": 할인X");
        }

        return price * discountPercent / 100;
    }
}
```

String 사용 시 타입 안정성 부족 문제

```java
class Main {
    public static void main(String[] args) {
        int price = 10000;

        DiscountService discountService = new DiscountService();

        // 존재하지 않는 등급
        int vip = discountService.discount("VIP", price);
        System.out.println("VIP 등급의 할인 금액: " + vip);

        // 오타
        int diamondd = discountService.discount("DIAMONDD", price);
        System.out.println("DIAMONDD 등급의 할인 금액: " + diamondd);

        // 소문자 입력
        int gold = discountService.discount("gold", price);
        System.out.println("gold 등급의 할인 금액: " + gold);
    }
}
```

```
/Users/qs/Library/Java/JavaVirtualMa
VIP: 할인X
VIP 등급의 할인 금액: 0
DIAMONDD: 할인X
DIAMONDD 등급의 할인 금액: 0
gold: 할인X
gold 등급의 할인 금액: 0
```

타입 안전 열거형 패턴

- 회원 등급을 다루는 클래스를 만들고, 회원 등급별로 상수를 선언
- 상수마다 별도의 인스턴스를 생성하고, 생성한 인스턴스를 대입
- 각각을 상수로 선언하기 위해 static , final 을 사용

```
public class ClassGrade {
    public static final ClassGrade BASIC = new ClassGrade();
    public static final ClassGrade GOLD = new ClassGrade();
    public static final ClassGrade DIAMOND = new ClassGrade();
}
```

```
class Main {
    public static void main(String[] args) {
        System.out.println("class BASIC = " + ClassGrade.BASIC.getClass());
        System.out.println("class GOLD = " + ClassGrade.GOLD.getClass());
        System.out.println("class DIAMOND = " + ClassGrade.DIAMOND.getClass());
        System.out.println();
        System.out.println("ref BASIC = " + ClassGrade.BASIC);
        System.out.println("ref GOLD = " + ClassGrade.GOLD);
        System.out.println("ref DIAMOND = " + ClassGrade.DIAMOND);
    }
}
```

```
/Users/qs/Library/Java/JavaVirtualMachines/o
class BASIC = class ClassGrade
class GOLD = class ClassGrade
class DIAMOND = class ClassGrade

ref BASIC = ClassGrade@3feba861
ref GOLD = ClassGrade@5b480cf9
ref DIAMOND = ClassGrade@6f496d9f
```

타입 안전 열거형 패턴의 적용

```
public class DiscountService {
    public int discount(ClassGrade classGrade, int price) {
        int discountPercent = 0;
        if (classGrade == ClassGrade.BASIC) {          discountPercent = 10;
        } else if (classGrade == ClassGrade.GOLD) {    discountPercent = 20;
        } else if (classGrade == ClassGrade.DIAMOND) {discountPercent = 30;
        } else {                                       System.out.println("할인X");}
        return price * discountPercent / 100;
    }
}
```

```
class Main {
    public static void main(String[] args) {
        int price = 10000;

        DiscountService discountService = new DiscountService();
        int basic = discountService.discount(ClassGrade.BASIC, price);
        int gold = discountService.discount(ClassGrade.GOLD, price);
        int diamond = discountService.discount(ClassGrade.DIAMOND, price);

        System.out.println("BASIC 등급의 할인 금액: " + basic);
        System.out.println("GOLD 등급의 할인 금액: " + gold);
        System.out.println("DIAMOND 등급의 할인 금액: " + diamond);
    }
}
```

```
BASIC 등급의 할인 금액: 1000
GOLD 등급의 할인 금액: 2000
DIAMOND 등급의 할인 금액: 3000
```

ClassGrade의 인스턴스 생성 문제

```
class Main {
    public static void main(String[] args) {
        int price = 10000;

        DiscountService discountService = new DiscountService();

        ClassGrade newClassGrade = new ClassGrade(); //생성자 private으로 막아야 함
        int result = discountService.discount(newClassGrade, price);
        System.out.println("newClassGrade 등급의 할인 금액: " + result);
    }
}
```

```
/Users/qs/Library/Java/JavaVirtualMachine
할인X
newClassGrade 등급의 할인 금액: 0
```

private 생성자

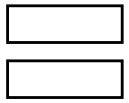
```
public class ClassGrade {
    public static final ClassGrade BASIC = new ClassGrade();
    public static final ClassGrade GOLD = new ClassGrade();
    public static final ClassGrade DIAMOND = new ClassGrade();

    private ClassGrade() {}
}
```

```
class Main {
    public static void main(String[] args) {
        int price = 10000;
        DiscountService discountService = new DiscountService();

        /*
        ClassGrade newClassGrade = new ClassGrade(); //생성자 private으로 막아야 함
        int result = discountService.discount(newClassGrade, price);
        System.out.println("newClassGrade 등급의 할인 금액: " + result);
        */
    }
}
```

열거형 - Enum Type

- 타입 안전 열거형 패턴(Type-Safe Enum Pattern)을 매우 편리하게 사용할 수 있는 **열거형(Enum Type)**을 제공

```
public class Grade extends Enum {
    public static final Grade BASIC = new Grade();
    public static final Grade GOLD = new Grade();
    public static final Grade DIAMOND = new Grade();

    //private 생성자 추가
    private Grade() {}
}
```



```
public enum Grade {
    BASIC, GOLD, DIAMOND
}
```

열거형 타입의 특성

```
class Main {
    public static void main(String[] args) {
        System.out.println("class BASIC = " + Grade.BASIC.getClass());
        System.out.println("class GOLD = " + Grade.GOLD.getClass());
        System.out.println("class DIAMOND = " + Grade.DIAMOND.getClass());
        System.out.println("ref BASIC = " + refValue(Grade.BASIC));
        System.out.println("ref GOLD = " + refValue(Grade.GOLD));
        System.out.println("ref DIAMOND = " + refValue(Grade.DIAMOND));
    }

    private static String refValue(Object grade) {
        return Integer.toHexString(System.identityHashCode(grade));
    }
}

enum Grade {
    BASIC, GOLD, DIAMOND
}
```

```
class BASIC = class Grade
class GOLD = class Grade
class DIAMOND = class Grade
ref BASIC = 3feba861
ref GOLD = 5b480cf9
ref DIAMOND = 6f496d9f
```

열거형 사용 코드

```
class Main {
    public static void main(String[] args) {
        int price = 10000;

        DiscountService discountService = new DiscountService();
        int basic = discountService.discount(Grade.BASIC, price);
        int gold = discountService.discount(Grade.GOLD, price);
        int diamond = discountService.discount(Grade.DIAMOND, price);

        System.out.println("BASIC 등급의 할인 금액: " + basic);
        System.out.println("GOLD 등급의 할인 금액: " + gold);
        System.out.println("DIAMOND 등급의 할인 금액: " + diamond);
    }
}

enum Grade { BASIC, GOLD, DIAMOND }

class DiscountService {
    public int discount(Grade grade, int price) {
        int discountPercent = 0;

        // Grade myGrade = new Grade(); // enum 생성 불가
        //enum switch 변경 가능
        if (grade == Grade.BASIC) { discountPercent = 10; }
        else if (grade == Grade.GOLD) { discountPercent = 20; }
        else if (grade == Grade.DIAMOND) { discountPercent = 30; }
        else { System.out.println("할인X");}
        return price * discountPercent / 100;
    }
}
```

열거형 – 주요 메서드

```java
import java.util.Arrays;

public class Main {
    public static void main(String[] args) {
        //모든 ENUM 반환
        Grade[] values = Grade.values();
        System.out.println("values = " + Arrays.toString(values));
        for (Grade value : values) {
            System.out.println("name=" + value.name() + ", ordinal=" + value.ordinal());
        }

        //String -> ENUM 변환, 잘못된 문자면 IllegalArgumentException 발생
        String input = "GOLD";
        Grade gold = Grade.valueOf(input);
        System.out.println("gold = " + gold); //toString() 오버라이딩 가능
    }
}

enum Grade {
    BASIC, GOLD, DIAMOND
}
```

```
values = [BASIC, GOLD, DIAMOND]
name=BASIC, ordinal=0
name=GOLD, ordinal=1
name=DIAMOND, ordinal=2
gold = GOLD
```

열거형 – 리팩토링1

```
public class DiscountService {
    public int discount(ClassGrade classGrade, int price) {
        int discountPercent = 0;
        if (classGrade == ClassGrade.BASIC) {          discountPercent = 10;
        } else if (classGrade == ClassGrade.GOLD) {    discountPercent = 20;
        } else if (classGrade == ClassGrade.DIAMOND) {discountPercent = 30;
        } else {                                       System.out.println("할인X");}
        return price * discountPercent / 100;
    }
}
```

```
class DiscountService {
    public int discount(ClassGrade classGrade, int price) {
        return price * classGrade.getDiscountPercent() / 100;
    }

    public static void main(String[] args) {
        int price = 10000;
        DiscountService discountService = new DiscountService();
        int basic = discountService.discount(ClassGrade.BASIC, price);
        int gold = discountService.discount(ClassGrade.GOLD, price);
        int diamond = discountService.discount(ClassGrade.DIAMOND, price);

        System.out.println("BASIC 등급의 할인 금액: " + basic);
        System.out.println("GOLD 등급의 할인 금액: " + gold);
        System.out.println("DIAMOND 등급의 할인 금액: " + diamond);
    }
}
```

열거형 – 리팩토링1

```
public class ClassGrade {
    public static final ClassGrade BASIC = new ClassGrade(10);
    public static final ClassGrade GOLD = new ClassGrade(20);
    public static final ClassGrade DIAMOND = new ClassGrade(30);

    private final int discountPercent;

    private ClassGrade(int discountPercent) {
        this.discountPercent = discountPercent;
    }

    public int getDiscountPercent() {
        return discountPercent;
    }
}
```

열거형 – 리팩토링2

```
enum Grade {
    BASIC(10), GOLD(20), DIAMOND(30);

    private final int discountPercent;

    Grade(int discountPercent) {
        this.discountPercent = discountPercent;
    }

    public int getDiscountPercent() {
        return discountPercent;
    }
}
```

```
public class DiscountService {
    public int discount(Grade grade, int price) {
        return price * grade.getDiscountPercent() / 100;
    }
}
```

열거형 – 리팩토링2

```
public class Main {

    public static void main(String[] args) {
        int price = 10000;

        DiscountService discountService = new DiscountService();
        int basic = discountService.discount(Grade.BASIC, price);
        int gold = discountService.discount(Grade.GOLD, price);
        int diamond = discountService.discount(Grade.DIAMOND, price);

        System.out.println("BASIC 등급의 할인 금액: " + basic);
        System.out.println("GOLD 등급의 할인 금액: " + gold);
        System.out.println("DIAMOND 등급의 할인 금액: " + diamond);
    }
}
```

열거형 – 리팩토링3

```
class DiscountService {
    public int discount(ClassGrade classGrade, int price) {
        return price * classGrade.getDiscountPercent() / 100;
    }

    public static void main(String[] args) {
        int price = 10000;
        DiscountService discountService = new DiscountService();
        int basic = discountService.discount(ClassGrade.BASIC, price);
        int gold = discountService.discount(ClassGrade.GOLD, price);
        int diamond = discountService.discount(ClassGrade.DIAMOND, price);

        System.out.println("BASIC 등급의 할인 금액: " + basic);
        System.out.println("GOLD 등급의 할인 금액: " + gold);
        System.out.println("DIAMOND 등급의 할인 금액: " + diamond);
    }
}
```

```
public class DiscountService {
    public int discount(Grade grade, int price) {
        return grade.discount(price);
    }
}
```

열거형 – 리팩토링3

```
public enum Grade {
    BASIC(10), GOLD(20), DIAMOND(30);

    private final int discountPercent;
    Grade(int discountPercent) { this.discountPercent = discountPercent;}
    public int getDiscountPercent() { return discountPercent; }

    //추가
    public int discount(int price) {
        return price * discountPercent / 100;
    }
}
```

```
public class Main {

    public static void main(String[] args) {
        int price = 10000;
        System.out.println("BASIC 등급의 할인 금액: " + Grade.BASIC.discount(price));
        System.out.println("GOLD 등급의 할인 금액: " + Grade.GOLD.discount(price));
        System.out.println("DIAMOND 등급의 할인 금액: " + Grade.DIAMOND.discount(price));
    }
}
```

➔ DiscountService 제거 가능

ENUM 목록의 추가

```
public class EnumRefMain3_4 {

    public static void main(String[] args) {
        int price = 10000;
        Grade[] grades = Grade.values();
        for (Grade grade : grades) {
            printDiscount(grade, price);
        }
    }

    private static void printDiscount(Grade grade, int price) {
        System.out.println(grade.name() + " 등급의 할인 금액: " + grade.discount(price));
    }
}

enum Grade {
    BASIC(10), BRONZE(15), GOLD(20), DIAMOND(30);
    private final int discountPercent;
    Grade(int discountPercent) { this.discountPercent = discountPercent;}
    public int getDiscountPercent() {
        return discountPercent;
    }
    //추가
    public int discount(int price) {
        return price * discountPercent / 100;
    }
}
```

System

시스템과 관련된 기본 기능들을 제공

```java
import java.util.Arrays;

public class SystemMain {

    public static void main(String[] args) {
        // 현재 시간(밀리초)를 가져온다.
        long currentTimeMillis = System.currentTimeMillis();
        System.out.println("currentTimeMillis = " + currentTimeMillis);

        // 현재 시간(나노초)를 가져온다.
        long currentTimeNano = System.nanoTime();
        System.out.println("currentTimeNano = " + currentTimeNano);

        // 환경 변수를 읽는다.
        System.out.println("getenv= " + System.getenv());

        // 시스템 속성을 읽는다.
        System.out.println("properties = " + System.getProperties());
        System.out.println("Java version: " + System.getProperty("java.version"));

        // 배열을 고속으로 복사한다.
        char[] originalArray = {'h', 'e', 'l', 'l', 'o'};
        char[] copiedArray = new char[5];
        System.arraycopy(originalArray, 0, copiedArray, 0, originalArray.length);

        // 배열 출력
        System.out.println("copiedArray = " + copiedArray);
        System.out.println("Arrays.toString = " + Arrays.toString(copiedArray));

        // 프로그램 종료
        System.exit(0);
    }
}
```

```
currentTimeMillis = 1730326044456
currentTimeNano = 432562332488458
getenv= {CONDA_PROMPT_MODIFIER=(base) , HOMEBREW_PREFIX=/opt/homebrew, GSETTINGS_SCHEMA_DIR_CONDA_BACKUP=, MANPATH=/opt/homebrew/share/man::, COMMAND_MODE=u
properties = {java.specification.version=23, sun.jnu.encoding=UTF-8, java.class.path=/Users/qs/Downloads/java-mid1/src/out/production/java-mid1, java.vm.ven
, java.vm.specification.vendor=Oracle Corporation, java.specification.name=Java Platform API Specification, apple.awt.application.name=SystemMain, sun.manag
Java version: 23.0.1
copiedArray = [C@7530d0a
Arrays.toString = [h, e, l, l, o]
```

Math

수학과 관련된 기능들을 제공

```java
public class Main {

    public static void main(String[] args) {
        // 기본 연산 메서드
        System.out.println("max(10, 20): " + Math.max(10, 20)); // 최대값
        System.out.println("min(10, 20): " + Math.min(10, 20)); // 최소값
        System.out.println("abs(-10): " + Math.abs(-10)); // 절대값

        // 반올림 및 정밀도 메서드
        System.out.println("ceil(2.1): " + Math.ceil(2.1)); // 올림
        System.out.println("floor(2.1): " + Math.floor(2.1)); // 내림
        System.out.println("round(2.5): " + Math.round(2.5)); // 반올림

        // 기타 유용한 메서드
        System.out.println("sqrt(4): " + Math.sqrt(4)); //제곱근
        System.out.println("random(): " + Math.random()); //0.0 ~ 1.0 사이의 double 값
    }
}
```

```
/Users/qs/Library/Java/JavaVirtualMachines
max(10, 20): 20
min(10, 20): 10
abs(-10): 10
ceil(2.1): 3.0
floor(2.1): 2.0
round(2.5): 3
sqrt(4): 2.0
random(): 0.7719167283349427
```

Random

임의의 값과 관련된 기능들을 제공

```java
import java.util.Random;

public class RandomMain {

    public static void main(String[] args) {
        Random random = new Random();
//        Random random = new Random(1); //seed가 같으면 Random의 결과가 같다.

        int randomInt = random.nextInt();
        System.out.println("randomInt: " + randomInt);
        System.out.println("randomInt: " + random.nextInt());

        double randomDouble = random.nextDouble();//0.0d ~ 1.0d
        System.out.println("randomDouble: " + randomDouble);

        boolean randomBoolean = random.nextBoolean();
        System.out.println("randomBoolean: " + randomBoolean);

        // 범위 조회
        int randomRange1 = random.nextInt(10);//0 ~ 9까지 출력
        System.out.println("0 ~ 9: " + randomRange1);

        int randomRange2 = random.nextInt(10) + 1;// 1 ~ 10까지 출력
        System.out.println("1 ~ 10: " + randomRange2);
    }
}
```

```
randomInt: 357101820
randomInt: -1959952230
randomDouble: 0.3649022615542199
randomBoolean: false
0 ~ 9: 5
1 ~ 10: 9
```