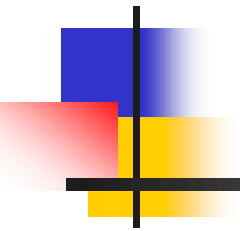


Interface 연습

경복대학교
소프트웨어융합과
배희호 교수

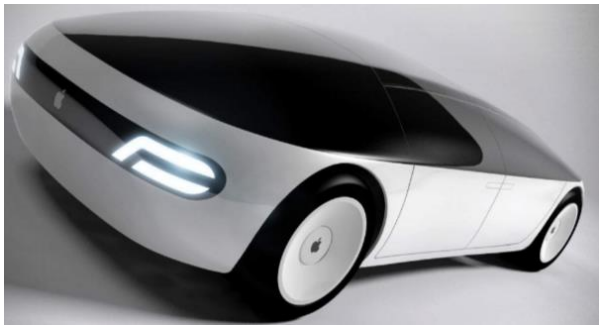
자동차 세금

- 탈것(vehicle)은 자동차(car)와 오토바이(motorcycle)로 구분하여 다음과 같이 세금(Tax)을 계산한다
  - 탈것은 금액에 따라 일정 금액의 세금을 납부함
  - 탈것은 기본 세금에 다음 조건에 따라 교육 세금(education tax)과 환경 세금(environmental tax)을 추가로 부과함
    - 연료를 가솔린(Gasoline)이나 전기(Electricity)를 사용하는 자동차는 환경 세금을 부과하지 않음
    - 연료를 디젤(Diesel)을 사용하는 자동차는 환경 세금을 추가로 부과함
    - 오토바이는 교육 세금을 부과하지 않음

자동차 세금

기본 세금
(10년 초과는 가중)

환경 세금



환경 세금
(연료: 디젤)

교육 세금

자동차 세금

- 처리 조건
  - 차종 코드

Type 1	Type 2	Type 3
승용차	승합차	오토바이

  - 연료 종료 코드

Type 1	Type 2	Type 3
Gasoline	Diesel	Electricity

  - 오토바이는 연료로 Diesel만을 사용함

자동차 세금

- 처리 조건
  - 자동차의 기본 세금은 가격에 따라 차등 부과

자동차 가격	세율
1,800,000이하	0.7%
1,800,000초과 3,600,000이하	0.8%
3,600,000초과	0.9%

  - 제조 년도가 10년이 초과한 자동차는 기본 세금의 초과한 년도의 비율로 추가 세금을 할증 부과함
    - 예) 만약 11년이 초과되었으면 기본 세금의 11%를 추가로 기본 세금에 할증함

자동차 세금

- 처리 조건
  - 오토바이의 기본 세금은 배기량에 따라 차등 부과

배기량	세율
90CC이하	0.5%
90CC초과 180CC이하	0.6%
180CC초과	0.7%

  - 제조 년도가 10년이 초과한 오토바이는 기본 세금의 초과한 년도의 비율로 추가 세금을 할증 부과함
    - 만약 11년이 초과되었으면 기본 세금의 11%를 추가로 기본 세금에 할증함

자동차 세금

- 처리 조건
  - 환경(Environment) 세금은 연료를 Diesel을 사용하는 탈 것의 기본 세금의 7%를 부과 함
  - 교육(Education) 세금은 자동차이면 기본 세금의 10%를 부과 하고, 오토바이는 기본 세금의 11%를 부과 함

자동차 세금

실행 결과

소유주	모델	제조사	년도	차종	연료	배기량	가격	세금	교육세	환경세	납부세액
정통파	SM9	삼성자동차	2021년	승용차	Electric	3,000CC	48,700,000	438,300	43,830	0	482,130
한민국	SM9	삼성자동차	2020년	승용차	Gasoline	3,000CC	46,700,000	420,300	42,030	0	462,330
자동차	SportsAge	기아자동차	1999년	승합차	Diesel	2,500CC	28,700,000	320,292	32,029	22,420	374,741
친환경	SportsAge	기아자동차	1999년	승합차	Gasoline	2,500CC	28,700,000	320,292	32,029	0	352,321
경복대	SportsAge	기아자동차	2011년	승합차	Diesel	2,500CC	28,700,000	289,296	28,929	20,250	338,475
홍길동	SONATA	현대자동차	2019년	승용차	Gasoline	1,990CC	27,500,000	247,500	24,750	0	272,250
바이든	Street10	할데이비슨	2009년	바이크	Diesel	450CC	27,800,000	221,844	0	24,402	246,246
이여주	Auto10	효성모터스	2010년	바이크	Diesel	650CC	17,800,000	140,798	0	15,487	156,285
이동국	Auto10	효성모터스	2020년	바이크	Diesel	150CC	17,800,000	106,800	0	11,748	118,548
트럼프	Street10	할데이비슨	2014년	바이크	Diesel	450CC	7,800,000	54,600	0	6,006	60,606

현재 날짜와 시간 구하기

- Calendar 클래스 활용 방법
  - '현재' 날짜와 시간을 구하기 위해 단 하나의 싱글톤 객체를 생성
  - getInstance() 메소드 사용
  - get() 메소드로 필요한 부분만 얻어 int 타입 변수 저장

```
Calendar cal = Calendar.getInstance();
int year = cal.get(Calendar.YEAR);
int month = cal.get(Calendar.MONTH) + 1;
int day = cal.get(Calendar.DAY_OF_MONTH);
int hour = cal.get(Calendar.HOUR_OF_DAY);
int min = cal.get(Calendar.MINUTE);
int sec = cal.get(Calendar.SECOND);
```

현재 날짜와 시간 구하기

- Date 클래스 활용 방법
  - 현재는 삭제 되었음
  - getYear() 메소드 사용
  - 1900년도 이후 값을 반환하므로 1900을 더해 주어야 함

```
Date time = new Date();
int year = time.getYear() + 1900;
int month = time.getMonth() + 1;
int day = time.getDate();
int hour = time.getHours();
int min = time.getMinutes();
int sec = time.getSeconds();
```

현재 날짜와 시간 구하기

- System.currentTimeMillis() 활용 방법
  - 1970년 1월 1일부터 경과한 시간을 long 값으로 리턴
  - 밀리 세컨드(1/1000초) 값을 리턴
  - SimpleDateFormat 클래스의 fomat() 메소드 활용

```
SimpleDateFormat format = new SimpleDateFormat ( "yyyy");
int year = Integer.parseInt(format.format(System.currentTimeMillis()));
```

자동차 세금

- TaxMethod Interface

```
public interface TaxMethod {
    public int tax();
    public int environment();
    public int education();

    default int total() {
        return tax() + education()+ environment();
    }
}
```

자동차 세금

- Vehicle 클래스

```
public abstract class Vehicle implements TaxMethod{
    private final String owner;
    private final String manufacturer;
    private final String model;
    private final int year;

    public Vehicle(String owner, String manufacturer, String model, int year) {
        this.owner = owner;
        this.manufacturer = manufacturer;
        this.model = model;
        this.year = year;
    }

    public int getYear() {
        return year;
    }
```

자동차 세금

- Vehicle 클래스

```
    @Override
    public String toString() {
        return String.format("%3s %10.9s %6s %4d년 ",
                                                owner, model, manufacturer, year);
    }
}
```

자동차 세금

- Vehicle을 상속한 Car 클래스

```
public class Car extends Vehicle {
    private final String type;
    private final String fuelType;
    private final int engineCapacity;
    private final int price;

    public Car(String owner, String manufacturer, String model, int year,
                                int price, char type, char fuelType, int engineCapacity) {
        super(owner, manufacturer, model, year);
        this.price = price;
        if (type == '1')
            this.type = "승용차";
        else if (type == '2')
            this.type = "승합차";
        else
            this.type = "미상";
```

자동차 세금

- Vehicle을 상속한 Car 클래스

```
        if (fuelType == '1')
            this.fuelType = "Gasoline";
        else if (fuelType == '2')
            this.fuelType = "Diesel";
        else if (fuelType == '3')
            this.fuelType = "Electricity";
        else
            this.fuelType = "미상";
        this.engineCapacity = engineCapacity;
    }

    @Override
    public int environment() {
        int environment = 0;
        if (fuelType.equals("Diesel"))
            environment = (int) (tax() * 7 / 100.0f);
        return environment;
    }
```

자동차 세금

- Vehicle을 상속한 Car 클래스

```
@Override
public int education() {
    return (int) (tax() * 10 / 100.0f);
}

@Override
public int tax() {
    int tax;
    if (price <= 1800000)
        tax = (int) (price * 0.7 / 100.0f);
    else if (price <= 3600000)
        tax = (int) (price * 0.8 / 100.0f);
    else
        tax = (int) (price * 0.9 / 100.0f);
```

자동차 세금

- Vehicle을 상속한 Car 클래스

```
        Calendar cal = Calendar.getInstance();
        int temp = cal.get(Calendar.YEAR) - getYear();
        if (temp > 10) {
            tax += (int) (tax * temp / 100.0f);
        }
        return tax;
    }

    @Override
    public String toString() {
        return super.toString() +
                String.format("%5.4s %9.8s %,5dCC %,11d %,9d %,7d %,7d %,8d",
                        type, fuelType, engineCapacity, price,
                                        tax(), education(), environment(), total());
    }
}
```

자동차 세금

- Vehicle을 상속한 MotorCycle 클래스

```
public class MotorCycle extends Vehicle {
    private final int engineCapacity;
    private final int price;

    public MotorCycle(String owner, String manufacturer, String model,
                                                    int year, int price, int engineCapacity) {
        super(owner, manufacturer, model, year);
        this.price = price;
        this.engineCapacity = engineCapacity;
    }

    @Override
    public int environment() {
        return  (int) (tax() * 11 / 100.0f);
    }
```

자동차 세금

```
@Override
public int education() {
    return 0;
}

@Override
public int tax() {
    int tax;
    if (engineCapacity <= 90)
        tax = (int)(price * 0.5f / 100.0f);
    else if (engineCapacity <= 180)
        tax = (int)(price * 0.6f / 100.0f);
    else
        tax = (int)(price * 0.7f / 100.0f);
    Calendar cal = Calendar.getInstance();
    int temp = cal.get(Calendar.YEAR) - getYear();
    if (temp > 10) {
        tax += (int)(tax * temp / 100.0f);
    }
    return tax;
}
```

자동차 세금

- Vehicle을 상속한 MotorCycle 클래스

```
    @Override
    public String toString() {
        return super.toString() +
                String.format("%5.4s %9.8s %,5dCC %,11d %,9d %,7d %,7d %,8d",
                        "바이크", "Diesel", engineCapacity, price,
                                                tax(), education(), environment(), total());
    }
}
```

자동차 세금

- 세금을 징수하는 TaxOffice 클래스

```
public class TaxOffice {
    private Vehicle[] vehicles;

    public TaxOffice(Vehicle[] vehicles) {
        this.vehicles = vehicles;
    }

    public void sort() {
        Vehicle temp;
        for (int i = 0; i < vehicles.length - 1; i++) {
            for (int j = i + 1; j < vehicles.length; j++) {
                if (vehicles[i].total() < vehicles[j].total()) {
                    temp = vehicles[j];
                    vehicles[j] = vehicles[i];
                    vehicles[i] = temp;
                }
            }
        }
    }
```

자동차 세금

- 세금을 징수하는 TaxOffice 클래스

```
    public void display() {
        sort();
        line();
        System.out.println("소유주     모델       제조사  년도     차종     연료
                    배기량      가격       세금    교육세  환경세  납부세액    ");
        line();
        for (int i = 0; i < vehicles.length; i++)
            System.out.println(vehicles[i]);
        line();
    }

    private void line() {
        System.out.println("**********************************************
                ************************************************");
    }
}
```

자동차 세금

- Main 클래스

```
public static void main(String[] args) {
    Vehicle[] car = {
        new Car("홍길동", "현대자동차",
                "SONATA", 2019, 27500000, '1', '1', 1990),
        new Car("경복대", "기아자동차",
                "SportsAge", 2011, 28700000, '2', '2', 2500),
        new Car("자동차", "기아자동차",
                "SportsAge", 1999, 28700000, '2', '2', 2500),
        new Car("친환경", "기아자동차",
                "SportsAge", 1999, 28700000, '2', '1', 2500),
        new MotorCycle("트럼프", "할데이비슨",
                "Street10", 2014, 7800000, 450),
        new MotorCycle("바이든", "할데이비슨",
                "Street10", 2009, 27800000, 450),
        new MotorCycle("이동국", "효성모터스",
                "Auto10", 2020, 17800000, 150),
```

자동차 세금

- Main 클래스

```
            new MotorCycle("이여주", "효성모터스",
                                        "Auto10", 2010, 17800000, 650),
            new Car("한민국", "삼성자동차",
                                       "SM9", 2020, 46700000, '1', '1', 3000),
            new Car("정통파", "삼성자동차",
                                      "SM9", 2021, 48700000, '1', '3', 3000)};

    TaxOffice tax = new TaxOffice(car);
    tax.display();
}
```