객체지향 프로그래밍 2차 과제

주차 관리 시스템

실습 교실 : 328호

실습 조교 : 이성빈 조교

학과 : 디지털 미디어 학과

학년 : 3학년

학번 : 201421105

이름: 김동해

1. 소개

2018-2학기 객체지향 프로그래밍 2차 과제의 내용은 주차 관리 시스템을 만드는 것이다. 주차 관리 시스템이란 차량의 종류, 차량의 무게, 차량의 탑승 인원을 입력받고 출차 시간을 입력받아 그에 맞는 주차 비용과 주차장 사용 시간을 출력해내는 프로그램이다. 나는 이 프로그램을 구현하는 데 가장 중요하다고 생각한 부분은 차량이 어떤 data와 method를 공유하고 어떤 data를 따로 쓰는지에 대한 것이 가장 중요하다고 생각했고 고민도 많이 했다. 만약 공유하는 data와 method가 있다면 Superclass에서 구현해 Override와 상속의 특징들을 이용해 구현해야 한다고 생각했다. 나는 총 11개의 Class를 만들었다. 그중에서도 중요하다고 생각하는 Class는 모든 차량의 SuperClass인 Vehicle class와 Time Class 그리고 Parking System Class가 중요하다고 생각했다. Vehicle Class는 모든 차량의 Super Class로 차량마다 가장 기본적인 data와 method가 정의되어 있다. Time Class는 차량마다 가지고 있는 시간 정보를 가지고 있다. 마지막으로 Parking System Class에는 배열에 각 차를 넣는 method들이 정의되어 있다. 나머지 class들은 각자 계산하는 방법에 대한 것이 정의되어있다.

2. 분석/설계

각 Class와 method에 대해 설명을 하겠다.

Vehicle Class

Vehicle Class는 모든 차량의 Super Class이며 Abstract Class이다. Super Class이기 때문에 각 차량에 기본적으로 쓰이는 Method와 Data를 가지고 있는 Class이며 Vehicle Class에서는 굳이 구현할 필요 없는 Method들을 추상 Method로 만들어 구현했다.

VehicleNumber. : int -> 각 차량의 차량 번호를 나타내는 Data

time : Time -> 각 차량의 입차 시간을 나타내며 후에 요금 계산에 사용

fair : int -> 주차 비용을 나타내는 Data

Vehicle Class

Public toString(); : abstract Method로 각 사용한 시간을 출력한다.

Public GetAdditionalCost(); : 기본 시간에 더 해진 시간에 대한 요금 출력

Public Vehicle() : Construct 부분으로 VehicleNumber와 Time을 초기화

Public getMethod() : 각 객체가 가지고 있는 데이터 출력

Public setMethod() : 각 객체가 가지고 있는 데이터를 변경할 때 사용

Public getFair() : 각 객체가 이용한 시간에 대한 요금을 출력한다.

Time Class

Time Class에 의해 만들어지는 객체는 각 차량 Class에서 데이터의 한 부분으로 쓰이고 주차장 사용 시간에 대한 정보를 가지고 있는 Class이다. Time Class에서는 년, 월, 일, 시, 분에 대한 Data를 가지고 있으며 이 정보는 처음 입차했을 때는 입차에 대한 정보가 되며 출차할 때는 입력 받은 출차 시간과 입차 시간을 빼 주차 비용을 계산할 때 사용한다. 각 Method들의 내용은 밑에 설명하겠다.

Time Class

year : int -> 연도를 의미한다.

month : int -> 월을 의미한다

day : int -> 일을 의미한다.

hour : int -> 시간을 의미한다.

minute : int -> 분을 의미한다.

Public Time() : Construct 부분으로 year, month, day, hour, minute을 초기화한다.

Public getMethod() : 각 객체가 가지고 있는 데이터 출력

Public setMethod() : 각 객체가 가지고 있는 데이터를 변경할 때 사용

Public InParking() : 차를 입차할 때 사용하며 차량 객체를 만들 때 data를 0으로 초기화 하여 만들고 이 method를 이용해 입차 시간을 Scanner로 입력 받는다 처음 만들 때부터 객체 입차 시간 정보를 입력 받는게 아니라 이 method를 이용해 변경해줘 입차 시간을 입력한다

Public OutParking() : 출차할 때 시간을 입력 받는 method이다. 이 method는 출차 시간만 입력 받는 것이 아니라 입력을 받았을 때 주차 30일이 넘지 않는지 출차 시간이 입차 시간보다 전에 시간인지를 판별한 후 이 2개의 사항에 불만족 할 때 입력이 허락 되도록 만들어진 method이다. 그 후에 입력 받고 달이 넘어가거나 시가 넘어갈 때도 고려해 총 주차장 이용 금액을 구하기 위한 정보를 출력하는 method이다.

Public OverTimeMinute() : 출차 시간에 error가 없고 분이 음수 일 때 시간에서 분을 얻어와 계산하는 method이다.

Public OverTimeHour() : 계산된 시간이 음수일 일 때 일에서 시간을 얻어오는 method이다.

Public OverTimeDay() : 계산된 일이 음수일 일 때 월에서 일을 얻어오는 method이다.

Public OverTimeMonth1() : 계산된 월(월 마지막 일이 31일)이 음수일 일 때 년에서 월을 얻어오는 method이다.

Public OverTimeMonth2() : 계산된 월(월 마지막 일이 28일)이 음수일 일 때 년에서 월을 얻어오는 method이다.

Public OverTimeMonth3() : 계산된 월(월 마지막 일이 30일)이 음수일 일 때 년에서 월을 얻어오는 method이다.

ParkingSystem Class

ParkingSystem Class는 Vehicle Class와 Time Class 두 개의 Class는 각 차량에 대한 시간 정보, 차량 정보를 가지고 있다면 ParkingSystem Class는 ArrayList에 차량을 넣는 method들로 이루어져 있다. 차의 종류에 따라 Method가 다르다.

Public SmallTruck BigTruckParking() : 위의 method의 방식과 같은 방법으로 SmallTruck의 객체를 만든 후에 SmallTruck 객체를 반환한다.

Public BigTruck BigTruckParking() : 위의 method의 방식과 같은 방법으로 BigTruck의 객체를 만든 후에 BigTruck 객체를 반환한다.

Public MiddleBus MiddleBusParking() : 위의 method의 방식과 같은 방법으로 MiddleBus의 객체를 만든 후에 MiddleBus 객체를 반환한다.

Public BigBus BigBusParking() : 위의 method의 방식과 같은 방법으로 BigBus의 객체를 만든 후에 BigBus 객체를 반환한다.

Public MiddleTruck MiddleTruckParking() : 위의 method의 방식과 같은 방법으로 MiddleTruck의 객체를 만든 후에 MiddleTruck 객체를 반환한다.

Public SmallBus SmallBusParking() : 위의 method의 방식과 같은 방법으로 SmallBus의 객체를 만든 후에 SmallBus 객체를 반환한다.

ParkingSystem Class

존재하지 않는다.

Public Car CarParking() : 먼저 차량번호를 입력 받고 0으로 초기화 한 Time 객체를 만든다. 이 2개의 Data를 가지고 Car 객체를 만든다. 그 후에 차의 Time 정보를 Time 객체의 InPakring method를 이용해 입차 시간을 입력받고 입차 시간 정보를 가지고 있는 차 객체를 반환한다.

나머지는 각 차량에 대한 Class이다. 하지만 Class 설명 부분에는 Car, Truck, Bus 로 나누어 3개만 설명 하겠다. 왜냐하면 Truck 종류하고 Bus 종류는 시간 처리 방법은 같고 가격만 다르기 때문에 굳이 설명할 필요가 없다고 생각한다.

Car Class

Car Class는 승용차를 의미하는 Class으로 Vehicle Class를 상속받은 Class이다. Car Class의 가장 중요한 부분은 시간에 따라 어떻게 주차 비용이 증가하는지 어떻게 시간이 정해지는 지가 중요한 부분이다.

Public void toString1() : 주차 시간을 나타내는 method로 CarClass에서는 분의 일의 자리가 0일 경우와 1일 경우 2가지로 나눠서 출력한다.

Public String toString() : 차량의 정보를 나타내느 method이며 차량 종류, 차량 번호, 입차 시간을 출력하는 method이다.

Public double getFair() : AdditionalCost와 BasicCost(기본 요금)을 합쳐서 총 주차 비용을 출력하는 method이다.

Public double GetAdditionalCost() : Car Class는 기본 30분에 1000원이며 그 후부터는 10분당 500원씩 추가되며 무조건 올림을 적용한다. 이 조건을 만족시키면서 30분 이후에 나오는 추가 비용을 계산하는 method이다. 만약 주차 시간이 30분 미만이면 0을 반환하고 분의 일의 자리가 0이면 십의 자리 – 3 (30분을 제외하고 시작한다는 의미) 십의 자리 \* 500을 AdditionalCost에 넣었다. 마지막으로 올림이 필요하다면 십의 자리 – 3을 한 다음에 +1 후 \* 500을해 올림을 표현했다. (-3을 하는 이유는 처음 30분 제외 시키고 계산하려고 했다.)

VehicleNumber. : int -> 각 차량의 차량 번호를 나타내는 Data

time : Time -> 각 차량의 입차 시간을 나타내며 후에 요금 계산에 사용

fair : int -> 주차 비용을 나타내는 Data

Car Class

Bus

Bus Class도 마찬가지로 Vehicle Class를 상속받은 Class이다. Bus Class도 시간 계산이 중요하다.

Bus Class

VehicleNumber. : int -> 각 차량의 차량 번호를 나타내는 Data

time : Time -> 각 차량의 입차 시간을 나타내며 후에 요금 계산에 사용

fair : int -> 주차 비용을 나타내는 Data

Public void toString1() : 주차 시간을 나타내는 method로 BusClass에서는 분의 단위가 0<분<30, 30<분<60, 1시간 초과이면서 분이 0인 경우 3가지로 나눠서 출력

Public double getFair() : AdditionalCost와 BasicCost(기본 요금)을 합쳐서 총 주차 비용을 출력하는 method이다.

Public String toString() : 차량의 정보를 나타내는 method이며 차량 종류, 차량 번호, 입차 시간을 출력하는 method이다.

Public double GetAdditionalCost() : Bus Class들 최초 1시간은 기본 시간이며 추가 30분당 돈이 증가한다. 모든 bus Class들은 다 이런 방식으로 돈이 오르며 크기에 따라 가격이 다르다. 나는 총 경우의 수를 4가지로 나눴다. 첫번째는 시간이 1시간 이하인 경우 반환을 0을 했고 2번째 1시간 이상이면서 분이 0<분<30인 경우 , 3번째 1시간 이상이면 30<분<60 경우 마지막으로 1시간 초과면서 분이 0인 경우 4가지로 나누어 돈 계산을 했다.

Truck Class

Truck Class도 마찬가지로 Vehicle Class를 상속받은 Class이다. Truck Class도 시간 계산이 중요하다.

Public void toString1() : 주차 시간을 나타내는 method로 Truck Class에서는 분의 자리가 0인 경우와 1<분<60 인 경우를 나눠서 출력해냈다.

Public String toString() : 차량의 정보를 나타내는 method이며 차량 종류, 차량 번호, 입차 시간을 출력하는 method이다.

Public double getFair() : AdditionalCost와 BasicCost(기본 요금)을 합쳐서 총 주차 비용을 출력하는 method이다.

Bus Class

VehicleNumber. : int -> 각 차량의 차량 번호를 나타내는 Data

time : Time -> 각 차량의 입차 시간을 나타내며 후에 요금 계산에 사용

fair : int -> 주차 비용을 나타내는 Data

Public double GetAdditionalCost() : Truck Class들 최초 1시간은 기본 시간이며 조금 이라도 지나면 1시간 단위로 올림되어 계산된다. 나는 이것을 3가지의 경우의 수로 나누어 주차 비용 계산을 했다. 첫번째는 주차 시간이 1시간 미만인 경우 2번째는 1시간이상이며 분 단위가 0<분<60인 경우 마지막으로 시간이 1시간 초과이며 분 단위가 0인 경우 이 3가지로 나눠 주차 비용 계산을 했다.

Main Class

Main Class는 실질직으로 주차 관리를 하는 Class이다. 나는 주차장을 따로 만들 때 ArrayList를 이용해 만들었다. ArrayList는 java에서 제공해주는 API로 가져다 사용만하면 되고 많은 기능이 Method로 구현 되어 있어 사용하기 쉽다. Main Class는 처음 시작할 때 ArrayList의 크기(주차장의 크기)를 입력받는다. 그 후에 5가지의 기능이 있으며 입차를 할 때는 ArrayList.add()를 사용했고 출차할 대 ArrayList.remove()를 사용했으며 ArrayList.get()을 많이 사용해 주차장을 만들었다.

각 Class에 대한 Class Diagram

Main

ParkingSystem의 객체를 만들어 ArrayList에 차 객체를 넣는다.

ParkingSystem

<

각 차의 객체를 만들고 그것을 반환한다.

Time

<

<

Vehicle

Time 의 객체를 차량의 Data로 이용한다.

SmallTruck

Car

MiddleTruck

BigTruck

SmallBus

BigBus

MiddleBus

3. 분석/설계(프로그램 주요 구현 부분 설명)

가장 까다로웠던 부분은 주차 비용을 계산하는 부분이다. 각 차의 종류마다 계산하는 방법이 다르고 시간에 따라 계산해야해서 힘들었다. 각 차의 종류마다 모든 경우의 수를 생각해 하나씩 경우를 생각해 돈을 계산했다. 각 차의 주차 비용 계산에 대한 것은 위에 설명했다. 그리고 또 어려웠던 부분은 시간 계산하는 부분이다. 시간 계산하는 부분은 FlowChart를 보고 설명 하겠다.

Time의 OutParking()부분(주차 비용을 계산하는 부분)

출차 시간 입력

입력 받은 출차 시간과 원래 입차 시간을 비교하여 오류 검출

오류 검출

원래 객체가 가지고 있는 달의 일수가 31일, 30일 28일 인지 비교

출차 시간 – 입차 시간

출차 시간 – 입차 시간

부호 넘기는 method

부호 넘기는 method

출차 시간 – 입차 시간

부호 넘기는 method

계산된 값 객체에 저장

계산된 값 객체에 저장

계산된 값 객체에 저장

OutParking()은 시간 계산에서 가장 중요한 method이다. OutParking method는 입력 받은 출차 시간이 30일을 초과하지 않는지, 출차 시간이 입차 시간 보다 전의 시간대가 아닌지를 먼저 비교한 후 제대로 입력되었으면 그 후에 그 달의 마지막 일수가 31일 인지, 30일 인지 28일인지를 비교해 나온 시간에 대한 총 비용을 계산할 수 있게 하는 method이다.

Main Class의 입차 부분

나는 주차 관리 시스템을 만들기 위해 ArrayList를 사용했다.

범위에 벗어난 숫자 입력

숫자 말고 다른 것 입력

1~5까지의 숫자 입력

Try Catch문구 오류 해결

1을 입력

차량 종류와 차량의 중량, 탑승인원 정보들을 입력 받는다.

다시 입력

잘못된 차량 정보 입력

다시 입력 받는다.

ParkingSystem method()

While문에 속해 있으므로 다시 wille문 실행

ArrayList.add() 사용으로 배열에 넣기

출차 부분

범위에 벗어난 숫자 입력

1~5 입력

숫자 말고 다른 것 입력

2을 입력

다시 입력

Try Catch문구 오류 해결

true

false

ArrayList.isEmpty

차량번호 입력

ArrayList에 차량번호 없을 경우

Array에 있는 모든 차량과 차량번호 비교

각 객체의 toString1() 실행

OutParking() 실행

객체의 getFair() 실행 후 total money에 넣는다.

ArrayList.remove() 실행으로 항목 지운다.

출차 부분에서는 고려해야할 부분이 많았다. 먼저 주차장에 차량이 있는지 먼저 확인을 해야 했고 두번째로 차량이 있을 경우에 차량 번호 입력을 받을 때 없는 차량 번호를 입력했을 경우를 도 고려해야 했기 때문이다. 2번의 if문을 지나고 난 후에야 드디어 출차 시간 입력과 주차 비용 계산을 할 수 있다.

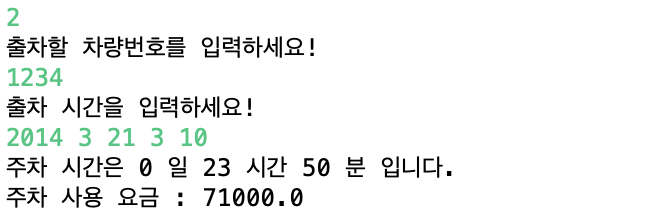
4. 실행 화면

1) 입차



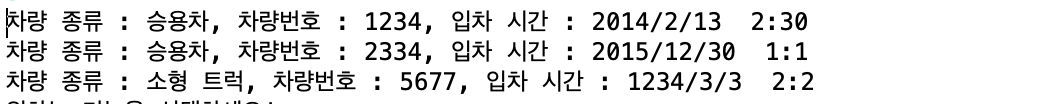
<차량을 종류 별로 입력>

2) 출차



<주차 시간에 대한 주차 비용 계산>

3) 주차장 차량 목록



<지금까지 입차 받은 차량 목록>

4) 하루 수입 계산



<지금까지 벌어들인 총 수입>

5) 프로그램 종료



<프로그램 종료>

5. 결론

1차 과제에서는 미니 장기를 구현했다. 미니 장기에서는 객체 사용법에 관해 공부했다면 2차 과제 주차 관리 시스템에서는 객체지향 프로그래밍의 가장 중요한 요소인 상속, 다형성을 사용해 프로그래밍 과제를 하는 것이었다. 상속의 중요한 개념은 Override와 Data를 상속받아 사용하는 것이다. 상속의 개념을 이용해 최대한 상속받은 클래스들이 공통으로 가지고 있는 data와 method들을 부모 클래스에 정의해 Override를 사용해 구현했다. 이 과제를 하면서 어려웠던 점 2가지가 있다. 첫 번째는 모든 차량이 공통으로 가지고 있는 Data와 사용해야 할 method들을 알아내는 것이었다. 처음에는 기본적인 Data와 method를 만들었고 그 후에 상속하면서 이건 공유해야 하는 Data와 method라고 생각하면 다시 SuperClass로 올라가 다시 구현하고 왔다 갔다 하면서 만들었다. 두 번째로 어려웠던 부분은 각기 다른 객체를 어떻게 한 배열에 넣는 것이었다. 그리고 추가하고 빼고 이런 기능을 어떻게 구현해야 하는 것 이였고 최근 수업에 배운 ArrayList를 이용해 만들었다. ArrayList는 add, remove,get등 많은 method들이 구현되어 있어 쉽게 사용할 수 있었다. 이 과제를 하면서 상속에 대한 것과 다형성에 대해 많이 배울 수 있었다.