Assignment 2 – Travel Scheduler

2020018159 이종준

0. 실행환경

1) OS: Windows 10

2) JAVA 버전: 15.0.2

3) IDE: VS Code

1. Activity.java, ExtremeActivity.java (extends Activity), ShowActivity.java (extends Activity)

1) constructor

public Activity (String name, String location, int price)

➔ name, location, price 를 받아 초기화. name 과 location 은 new 로 새로 할당해주었다.

public ExtremeActivity (String name, String location, int price, int minHeight, int minWeight)

public ShowActivity (String name, String location, int price, int minAge)

➔ 각각의 Activity 에서 필요한 추가 인자를 포함하여 구현. super(name, location, price)로 Acitivy 의 생성자를 먼저 호출하여 초기화해주었다.

public Activity (Activity otherActivity)

➔ copy constructor. name, location, price 를 otherActivity 의 getter 를 사용하여 deep copy 가 될 수 있도록 해주었다.

public ExtremeActivity (ExtremeActivity otherExtremeActivity)

public ShowActivity (ShowActivity otherShowActivity)

➔ copy constructor. 마찬가지로 super 생성자를 먼저 호출해준 뒤, 각각의 Activity 에서 추가적으로 갖고 있는 정보를 초기화해주었다.

2) getter, setter

➔ name 과 location 은 new 로 새로 할당한 String 을 return 할 수 있도록 하여 원본이 수정되는 일이 발생하지 않도록 해주었다.

3) toString, equals

➔ equals는 **Object obj가 null인지, getClass()를 이용하여 같은 클래스인지를 판단**한 후, 멤버 변수를 비교하도록 해주었다. toString은 명세의 출력 예시 형식에 맞추었다.

4) public int getActualPrice(Person person)

➔ 3 종류의 Activity (일반, Extreme, Show)를 통해 polymorphism 을 구현하기 위한 함수. 일반 Activity 에서는 인자로 받은 person 에 상관없이, **기본 가격을 return** 한다. **ExtremeActivity** 일 경우, person 의 나이가 **60 세 이상일 경우에만 30% 할증**하여 return 하고, 마지막으로 **ShowActivity** 일 경우, person 의 나이가 **19 세 이하일 경우에만 20% 할인**하여 가격을 return 한다.

2. Schedule.java

1) constructor

public Schedule (String name, int days, Person[] member)

➔ name(new 로 새로 할당), days 를 인자로 받아 초기화 해주고, plan 은 new Activity[12][days](**9~20 인 time 을 배열에서는 0~11 로 사용. day 역시 1~N 을 0~N-1 로 사용**)로, expense 는 0 으로 초기화해주었다. member 배열을 인자로 받아, this.member 배열에 deep copy 해주었다. **static 변수인 scheduleNum 은 생성자가 호출될 때마다 갱신될 수 있도록 ++ 처리해주었다.**

public Schedule (Schedule otherSchedule, String name)

➔ copy constructor. name 만 따로 받아 new 로 새로 할당하여 초기화하고, 나머지 정보들은 otherSchedule 에서 가져와 초기화. otherSchedule 의 plan 에서 activity 가 들어있으면, 해당 activity 를 deep copy 해주었고, activity 가 들어있지 않으면(null 이면) null 로 초기화해주었다. 이 과정에서 otherSchedule 에 포함된 activity 의 종류를 파악하여 따로 처리해주기 위해, getClass()를 이용하였다. otherSchedule 에 포함된

member 도 마찬가지로, deep copy 로 초기화 해주었다. **static 변수인 scheduleNum 은 마찬가지로 생성자가 호출될 때마다 갱신될 수 있도록 ++ 처리해주었다.**

2) addActivity throws InvalidAccessException, InsufficientConditionException

➔ 선택한 activity, day 와 time 을 인자로 받아 plan 에 추가. day 와 time 이 정상적인 범위로 입력되었는지, 해당 시간에 이미 activity 가 있는지, 동일한 activity 가 plan 에 이미 있는지를 먼저 파악함. 이 과정에서 해당 부분에 문제가 있다면, **InvalidAccessException 을 throw** 함. **입력 받은 activity 의 종류를 getClass()를 통해 파악**하여 따로따로 처리해주었다. 이 과정에서, Schedule 에 포함된 모든 멤버가 activity 의 조건에 부합하는지 체크하고, **만약 한 명이라도 조건에 부합하지 못한다면 InsufficientConditionException 을 throw 하도록 하였다.** expense 역시 person 과 activity 에 따라 달라져야 하므로, **getActualPrice()를 통해 전체 멤버의 price 를 각각 계산한 뒤 더해주었다.**

3) removeActivity throws InvalidAccessException

➔ day 와 time 을 인자로 받아 plan 에서 해당 시간대의 activity 를 제거함. day 와 time 이 정상적인 범위로 입력되었는지, 해당 시간대에 activity 가 있는지를 먼저 파악함. 이 과정에서 해당 부분에 문제가 있다면, **InvalidAccessException 을 throw** 함. plan 에서 해당 시간대를 null 로 변경해주고, **expense 를 차감할 때는 더할 때와 마찬가지로, getActualPrice()를 통해 전체 멤버의 price 를 각각 계산한 뒤 차감해 주었다.**

4) getter, setter

➔ Activity.java 와 마찬가지로, getName()같은 getter 에는 새로 할당하여 return 해주도록 new 로 새로 할당한 String 을 return 해주었음.

3. Person.java

1) constructor

public Person (String name, int age, int height, int weight)

➔ name, age, height, weight 를 받아 초기화. name 의 경우에는 new 로 새로 할당해주었다.

public Person (Person otherPerson)

➔ copy constructor. otherPerson 의 정보를 가져와 초기화. name 의 경우에는 마찬가지로 new 로 새로 할당해주었다.

2) toString

➔ 이름, 나이, 키, 몸무게 형식의 string 을 반환하도록 구현하였다.

3) getter, setter

➔ getName()에는 새로 할당하여 return 해주도록 new 로 새로 할당한 String 을 return 해주었음.

4. TravelScheduler.java

0) main 함수

➔ **FileInputStream 과 Scanner 를 이용하여, 기본 위치에 있는 ActivityList.txt 와 MemberList.txt 를 가져오도록 하였다 (./ActivityList.txt, ./MemberList.txt). 만약 파일이 없는 경우에는, 프로그램이 바로 종료되도록 하였다. 가져온 ActivityList 와 MemberList 를 통해, 새로 선언한 배열에 각각의 정보들을 초기화하였다 (split(“, ”)로 정보 구분). while 문 안에 try catch 구문을 넣어, exception 이 발생할 경우, 오류 메시지를 출력하고 계속해서 해당 동작을 반복하도록 하였다 (해당 로직은 이 Assignment 내 모든 exception handling 에 동일하게 사용되었다).**

1) printMainMenu throws InputMismatchException, InvalidAccessException

➔ 메인 메뉴를 출력하고, 입력을 받아 선택한 메뉴의 번호 (1,2,3)를 return 한다. 해당 과정에서 int 가 아닌 입력을 받을 경우, 1,2,3 의 범위를 벗어난 입력을 받을 경우 exception 을 throw 하도록 하였다. 해당 exception 은 메인 함수에서 catch 하도록 하였다.

2) selectSchedule

➔ 메인 메뉴에서 1 번을 누를 경우 이 함수를 호출. ScheduleList 를 출력하고(printScheduleList()), 입력을 받아 선택한 schedule 에서 어떤 작업을 할 것인지 세부 메뉴를 출력한다. **세부 메뉴는 명세 내용에 따라, while 문으로 무한 반복하며, 0 을 누르거나 EMPTY 를 누르면 메인 메뉴로 이동하도록 해주었다.** 그 이외의 잘못된 입력의 경우에는 exception 으로 다루도록 하였다. 1 번을 누르면 activity 를

추가(addActivity()), 2 번을 누르면 activity 를 삭제(removeActivity()), 그리고 3 번을 누르면 schedule 을 출력한다(printSchedule()).

3) printActivity

➔ activity 의 종류를 형식에 맞게 출력해주는 함수.

4) addActivity

➔ 추가하려는 activity 의 종류, day, time 을 입력받고, 앞선 단계에서 선택한 Schedule 객체의 addActivity 함수를 호출한다.

5) printSchedule

➔ 출력 예시의 형식에 맞게 schedule 의 plan 과 expense, 그리고 schedule 에 포함되어 있는 member 정보를 출력해준다. 앞선 단계에서 선택한 schedule 객체의 day 에 맞게 표를 작성해서 출력해주었으며, **printf 에서 "%-16s"를 사용하여 정렬해주었다.**

6) removeActivity

➔ 삭제하려는 activity 의 day 와 time 을 입력받고, 앞선 단계에서 선택한 Schedule 객체의 removeActivity 함수를 호출한다.

7) editSchedule

➔ 메인 메뉴에서 2 번을 누를 경우 이 함수를 호출. 세부 메뉴를 출력하고, 입력을 받아 세부 메뉴를 실행할 수 있도록 함. **Schedule 클래스의 scheduleNum 이 5 보다 작은 지 (추가할 수 있는 상태인지) 확인함.** 0 을 누를 경우 이전 단계로 돌아가게 해 주었음. 1 번을 누를 경우, schedule 의 이름과 총 days, 그리고 member 를 입력 받아 새로 schedule 객체를 생성하여 scheduleList 에 추가. **이때, 명세에는 나와있지 않지만, 올바른 일정표 출력을 위해 days 가 0 인 경우에 InvalidAccessException 을 throw 하도록 하였음. member 의 경우에는, 여러 명을 선택할 수 있으며, ", "로 입력을 구분하였음.** 2 번을 누를 경우, scheduleList 에서 copy 하려는 schedule 의 번호와 새로운 schedule 의 name 을 입력 받아 Schedule 클래스의 copy constructor 를 호출. **명세에는 나와있지 않지만, 만약 0 을 누를 경우, 메인 메뉴로 이동하도록 해 주었음 (scheduleList 가 비어 있는 상태에서 해당 메뉴를 들어갈 경우, 빠져나오지 못하는 문제를 해결하기 위해).**

8) printScheduleList

➔ scheduleList 를 출력함. 안에 schedule 객체가 들어 있다면 해당 schedule 의 name 을 출력하고, 비어 있다면 EMPTY SCHEDULE 을 출력하도록 함.

5. Exceptions

(InsufficientConditionException.java, InvalidAccessException.java, ArrayFullException.java)

➔ Exception 을 상속받음. 기본 생성자는 각각의 지정된 오류문구를 super()로 초기화하고, message 를 인자로 받는 생성자는 전달받은 message 를 super()로 초기화한다.