Assignment 1 – Travel Scheduler

2020018159 이종준

1. Activity.java

   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 Activity (Activity otherActivity)

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

   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 obj 가 null 인지, getClass()를 이용하여 같은 클래스인지를 판단**한 후, 멤버 변수를 비교하도록 해주었다. toString 은 명세의 출력 예시 형식에 맞추었다.

2. Schedule.java

   1) constructor

   public Schedule (String name, int days)

   ➔ 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. **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. **static 변수인 scheduleNum 은 마찬가지로 생성자가 호출될 때마다 갱신될 수 있도록 ++ 처리해주었다.**

2) addActivity

➔ 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. plan 에 새로운 2 차원 배열을 할당해주고, expense 역시 갱신해주었음.

3) removeActivity

➔ 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. plan 에서 해당 시간대를 null 로 변경해주고, expense 역시 갱신해주었음.

4) getter, setter

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

3. **TravelScheduler.java**

1) printMainMenu

➔ 메인 메뉴 출력. while 문 안에서 계속해서 반복하여 메인 메뉴를 출력하고, 입력을 받아 선택한 메뉴의 함수를 호출해준다.

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 누르면 메인 메뉴로 이동하도록 해주었다.** 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. 추가하려는 activity 의 종류를 입력할 때 0 을 누르게 되면 뒤로 가게 해주었다.

5) printSchedule

➔ 출력 예시의 형식에 맞게 schedule 의 plan 과 expense 를 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. 0 을 누를 경우 이전 단계로 돌아가게 해주었음. 1 번을 누를 경우, **Schedule 클래스의 scheduleNum 이 5 보다 작을 경우(추가할 수 있는 상태)**, schedule 의 이름과 메인 메뉴에서 1 번을 누를 경우 이 함수를 호출.총 days 를 입력받아 새로 schedule 객체를 생성하여 scheduleList 에 추가. **name 을 입력받는 과정에서 버퍼 문제로 인해 nextLine()이 올바르게 작동하지 못할 수 있어, nextLine()을 먼저 한 번 실행하여 버퍼를 비워주었음.** 2 번을 누를 경우, scheduleList 에서 copy 하려는 schedule 의 번호와 새로운 schedule 의 name 을 입력받아 Schedule 클래스의 copy constructor 를 호출. 만약 0 을 누르거나, EMPTY SCHEDULE 상태의 번호를 입력할 경우, 메인 메뉴로 이동하도록 해주었음.

8) printScheduleList

➔ scheduleList 를 출력함. 안에 schedule 객체가 들어있다면 해당 schedule 의 name 을 출력하고, 비어있다면 EMPTY SCHEDULE 을 출력하도록 함.