프로그램 실행 설명

0, empty schedule 고르면 처음으로

① Select schedule -> scheduleList 1~5 출력 / 숫자 고르기

-> ① Add activity -> activityList 1~8 출력 / 숫자고르기
날짜, 시간 고르기

② Remove activity -> 전체 스케쥴표와 가격 출력
삭제할 날짜, 시간 고르기

③ Print schedule -> 전체 스케쥴표와 가격 출력

무한반복

0 입력받으면 처음으로

3 입력받기 전까지 무한 반복

② Edit schedule -> ① Make a new schedule -> 이름, 전체 일 수 받고 schedule 생성

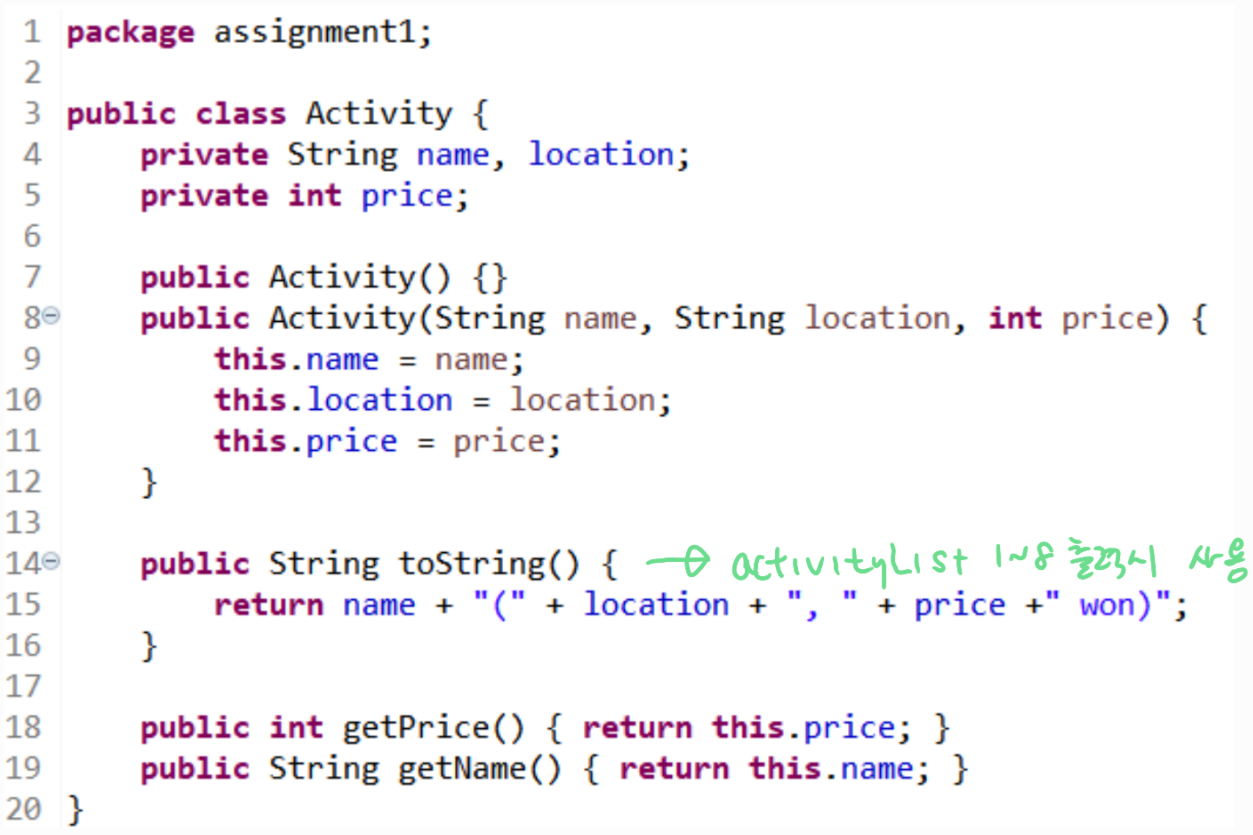
② Copy an existing sch -> scheduleList 1~5 출력 / 숫자, 이름 고르기

무한반복

0입력 받으면 처음으로

③ End program -> 프로그램 종료

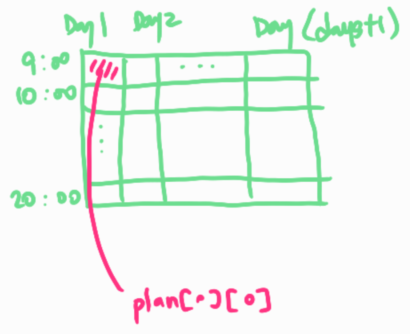
Class Activity



```java
package assignment1;

public class Activity {
    private String name, location;
    private int price;

    public Activity() {}
    public Activity(String name, String location, int price) {
        this.name = name;
        this.location = location;
        this.price = price;
    }

    public String toString() { → activityList 1~8 출력시 사용
        return name + "(" + location + ", " + price +" won)";
    }

    public int getPrice() { return this.price; }
    public String getName() { return this.name; }
}
```

Class Schedule

```java
package assignment1;

public class Schedule {
    private String name;
    private int days, expense;
    private Activity[][] plan;
    public static int scheduleNum;

    public Schedule(String name, int days) {
        this.name = name;
        this.days = days;
        this.expense = 0;
        plan = new Activity[days][12];
        scheduleNum++;
    }

    public Schedule(String name, Schedule s1) {
        this.name = name;
        this.days = s1.days;
        this.expense = s1.expense;
        this.plan = new Activity[days][12];
        for (int i = 0; i < days; i++)
            for (int j = 0; j < 12; j++)
                this.plan[i][j] = s1.plan[i][j];
        scheduleNum++;
    }

    public String getName() {
        return this.name;
    }

    public int getExpense() {
        return this.expense;
    }

    public int setPlan(Activity activity, int day, int time) {
        if (day > this.days || plan[day - 1][time - 9] != null)
            return 0;
        this.plan[day - 1][time - 9] = activity;
        this.expense += activity.getPrice();
        return 1;
    }
    public int removePlan(int day, int time) {
        if (day > this.days || plan[day - 1][time - 9] == null)
            return 0;
        this.expense -= this.plan[day - 1][time - 9].getPrice();
        this.plan[day - 1][time - 9] = null;
        return 1;
    }

    public String print(int day, int time) {
        if (plan[day - 1][time - 9] == null)
            return "----";
        else
            return this.plan[day - 1][time - 9].getName();
    }

    public void printSchedule() {
        for (int i = 1; i <= days; i++)
            System.out.print("------------------");
        System.out.println();
        System.out.print("                ");
        for (int i = 1; i <= days; i++)
            System.out.printf("%-16s", "Day " + i);
        System.out.println();
        for (int i = 0; i < 12; i++) {
            System.out.printf("%-16s", i + 9 + ":00");
            for (int j = 1; j <= days; j++)
                System.out.printf("%-16s", print(j, i + 9));
            System.out.println();
        }
        for (int i = 1; i <= days; i++)
            System.out.print("------------------");
        System.out.println();
        System.out.println("Total expenses: " + getExpense() + " won");
        for (int i = 1; i <= days; i++)
            System.out.print("------------------");
        System.out.println();
    }
}
```



서로 영향 못주게 새로운 주소 할당후 내용만 복사

→ 해당 날짜가 존재하지 않거나 이미 activity 있으면 plan을 추가하지 않음

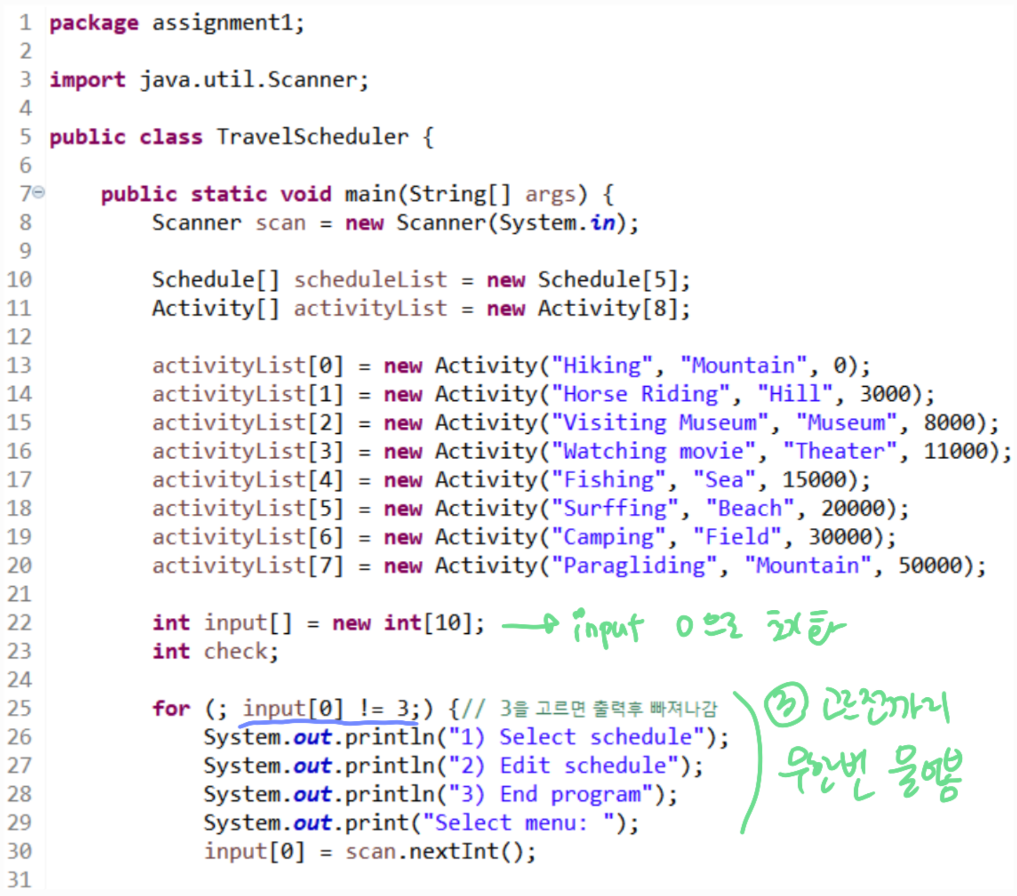
→ 해당 날짜에 시간에 activity가 없으면 remove 하지 않음

→ 해당 날짜에 시간에 activity가 없으면 ---- 출력, 있으면 activity 이름 출력

```
------------------------------------
                Day 1           Day 2
9:00            ----            ----
10:00           ----            ----
11:00           ----            ----
12:00           ----            ----
13:00           ----            ----
14:00           ----            ----
15:00           ----            ----
16:00           ----            ----
17:00           ----            ----
18:00           ----            ----
19:00           ----            ----
20:00           ----            ----
------------------------------------
Total expenses: 0 won
------------------------------------
```

출력

Class TravelScheduler



```java
package assignment1;

import java.util.Scanner;

public class TravelScheduler {

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        Schedule[] scheduleList = new Schedule[5];
        Activity[] activityList = new Activity[8];

        activityList[0] = new Activity("Hiking", "Mountain", 0);
        activityList[1] = new Activity("Horse Riding", "Hill", 3000);
        activityList[2] = new Activity("Visiting Museum", "Museum", 8000);
        activityList[3] = new Activity("Watching movie", "Theater", 11000);
        activityList[4] = new Activity("Fishing", "Sea", 15000);
        activityList[5] = new Activity("Surffing", "Beach", 20000);
        activityList[6] = new Activity("Camping", "Field", 30000);
        activityList[7] = new Activity("Paragliding", "Mountain", 50000);

        int input[] = new int[10];
        int check;

        for (; input[0] != 3;) {// 3을 고르면 출력후 빠져나감
            System.out.println("1) Select schedule");
            System.out.println("2) Edit schedule");
            System.out.println("3) End program");
            System.out.print("Select menu: ");
            input[0] = scan.nextInt();

```

$\rightarrow$ input 0으로 최화

③ 고르전까지 무한번 물어봄

```java
switch (input[0]) {
case 1:// 1) Select schedule
```

① Select Schedule 선택

```java
        // 만들어진 schedule를 나열
    for (int i = 0; i < scheduleList.length; i++) {
        if (scheduleList[i] != null)
            System.out.println(i + 1 + ") " + scheduleList[i].getName());
        else
            System.out.println(i + 1 + ") EMPTY SCHEDULE");
    }
    System.out.print("Select a schedule: ");
    input[1] = scan.nextInt();
    if (input[1] == 0 || scheduleList[input[1] - 1] == null) // 0 또는 EMPTY SCHEDULE을 선택하면 이전 메뉴로 돌아 감
        continue;
    // Schedule을 선택하면 해당 schedule에 대한 수정 및 출력을 반복 수행
    do {// 0고르면 탈출
        System.out.println("1) Add activity");
        System.out.println("2) Remove activity");
        System.out.println("3) Print schedule");
        System.out.print("Select menu: ");
        input[2] = scan.nextInt();
        switch (input[2]) {
        case 0:
            break;
        case 1:// input[1]에 add activity
            for (int i = 0; i < activityList.length; i++)
                System.out.println(i + 1 + ") " + activityList[i].toString());
            System.out.print("Select activity to do: ");
            input[3] = scan.nextInt();
            System.out.print("Enter the day to do activity: ");
            input[4] = scan.nextInt();
            System.out.print("Enter the time to do activity(9~20): ");
            input[5] = scan.nextInt();
            check = scheduleList[input[1] - 1].setPlan(activityList[input[3] - 1], input[4], input[5]);
            if (check == 0)
                System.out.println("Fail to add activity");
```

) 이미 있는 경우 / 스케줄범위 밖인 경우

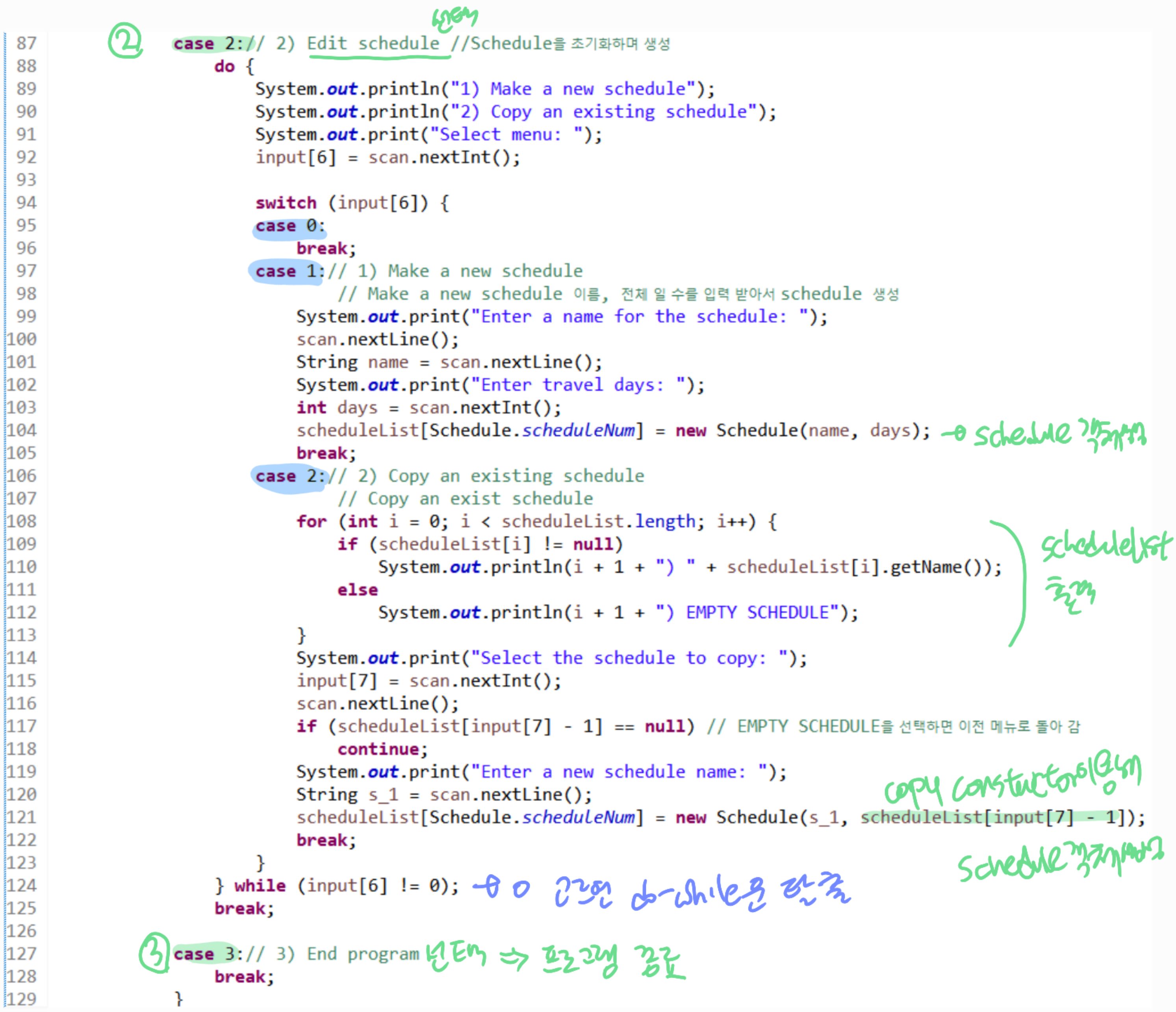
```java
            break;

        case 2:// input[1]에 remove activity
            scheduleList[input[1] - 1].printSchedule();
            System.out.print("Enter the day to remove activity: ");
            int day = scan.nextInt();
            System.out.print("Enter the time to remove activity: ");
            int time = scan.nextInt();
            check = scheduleList[input[1] - 1].removePlan(day, time);
            if (check == 1)
                System.out.println("Removed successfully");
```

plan이 없거나 / 스케줄범위 밖인 경우는 출력X

```java
            break;

        case 3:// input[1]에 print schedule
            scheduleList[input[1] - 1].printSchedule();
            break;
        }
    } while (input[2] != 0);
```

→ 0 고르면 do-while을 탈출

```java
    break;
```



```java
    case 2:// 2) Edit schedule //Schedule을 초기화하며 생성
        do {
            System.out.println("1) Make a new schedule");
            System.out.println("2) Copy an existing schedule");
            System.out.print("Select menu: ");
            input[6] = scan.nextInt();

            switch (input[6]) {
            case 0:
                break;
            case 1:// 1) Make a new schedule
                    // Make a new schedule 이름, 전체 일 수를 입력 받아서 schedule 생성
                System.out.print("Enter a name for the schedule: ");
                scan.nextLine();
                String name = scan.nextLine();
                System.out.print("Enter travel days: ");
                int days = scan.nextInt();
                scheduleList[Schedule.scheduleNum] = new Schedule(name, days);
                break;
            case 2:// 2) Copy an existing schedule
                    // Copy an exist schedule
                for (int i = 0; i < scheduleList.length; i++) {
                    if (scheduleList[i] != null)
                        System.out.println(i + 1 + ") " + scheduleList[i].getName());
                    else
                        System.out.println(i + 1 + ") EMPTY SCHEDULE");
                }
                System.out.print("Select the schedule to copy: ");
                input[7] = scan.nextInt();
                scan.nextLine();
                if (scheduleList[input[7] - 1] == null) // EMPTY SCHEDULE을 선택하면 이전 메뉴로 돌아 감
                    continue;
                System.out.print("Enter a new schedule name: ");
                String s_1 = scan.nextLine();
                scheduleList[Schedule.scheduleNum] = new Schedule(s_1, scheduleList[input[7] - 1]);
                break;
            }
        } while (input[6] != 0);
        break;

    case 3:// 3) End program
        break;
}
```

②

선택

→ schedule 객체생성

scheduleList 출력

copy constuctor이용해 schedule 객체생성

→ 0 고르면 do-while문 탈출

③ 선택 ⇒ 프로그램 종료