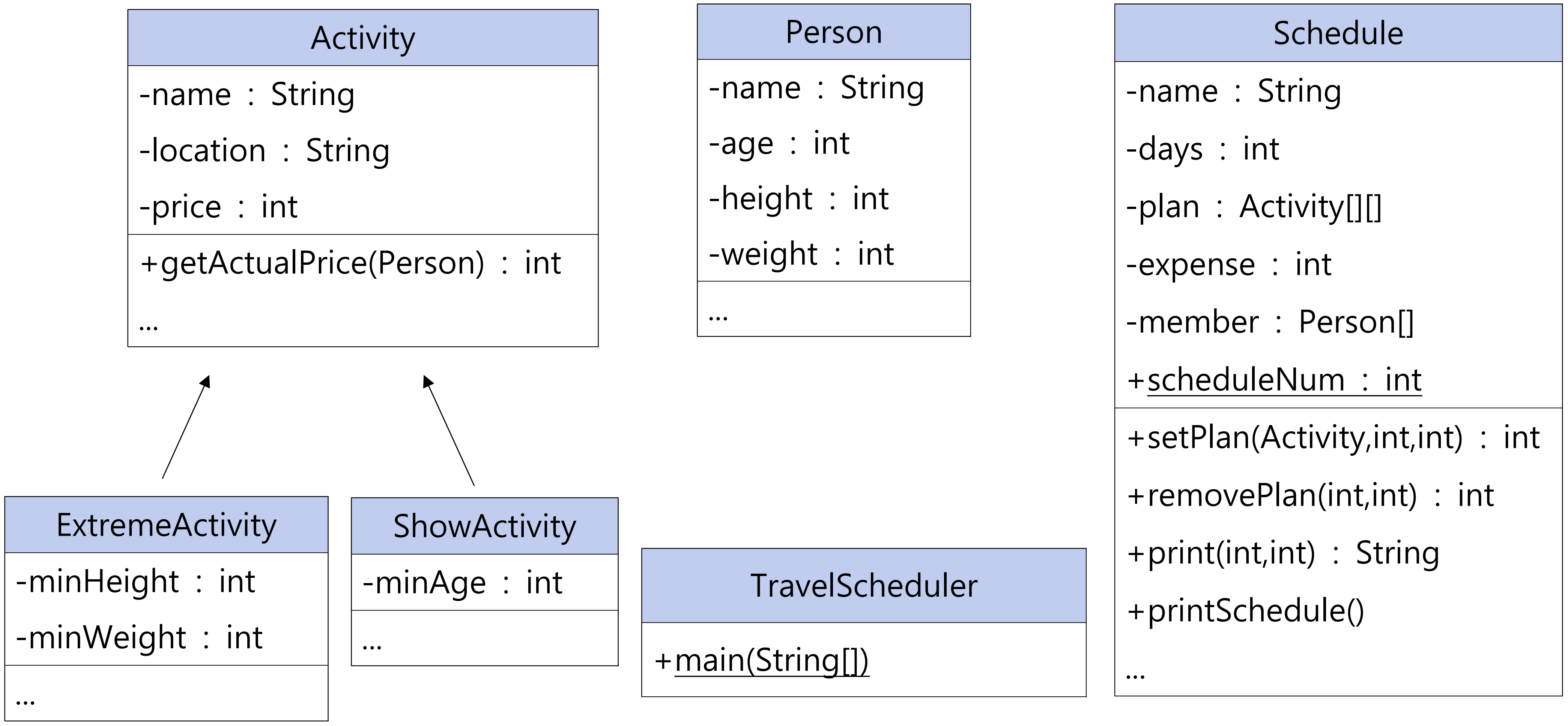
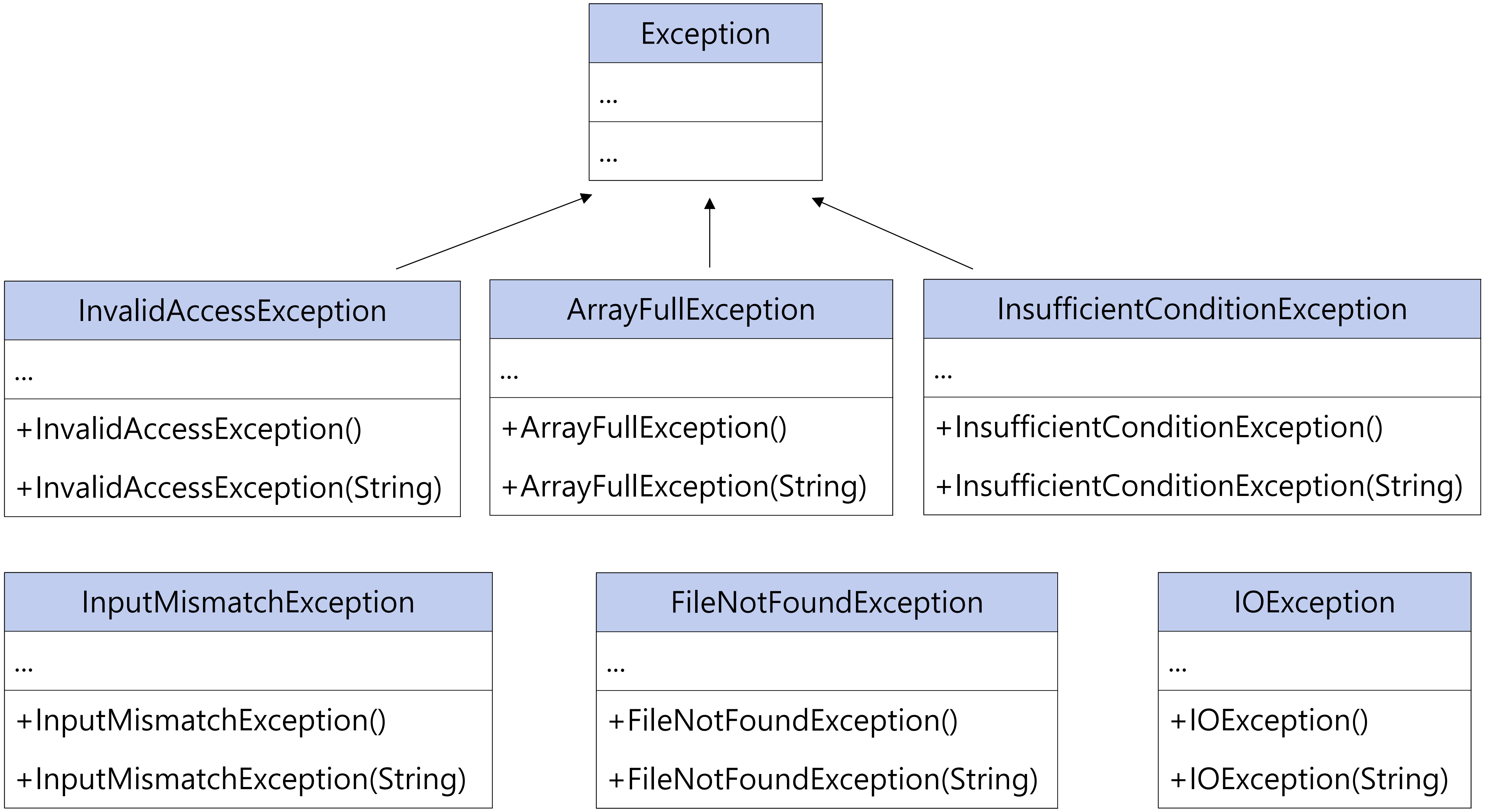
UML


Activity
-name : String
-location : String
-price : int
+getActualPrice(Person) : int
...
Person
-name : String
-age : int
-height : int
-weight : int
...
Schedule
-name : String
-days : int
-plan : Activity[][]
-expense : int
-member : Person[]
+scheduleNum : int
+setPlan(Activity,int,int) : int
+removePlan(int,int) : int
+print(int,int) : String
+printSchedule()
...
ExtremeActivity
-minHeight : int
-minWeight : int
...
ShowActivity
-minAge : int
...
TravelScheduler
+main(String[])


Exception
...
...
InvalidAccessException
...
+InvalidAccessException()
+InvalidAccessException(String)
ArrayFullException
...
+ArrayFullException()
+ArrayFullException(String)
InsufficientConditionException
...
+InsufficientConditionException()
+InsufficientConditionException(String)
InputMismatchException
...
+InputMismatchException()
+InputMismatchException(String)
FileNotFoundException
...
+FileNotFoundException()
+FileNotFoundException(String)
IOException
...
+IOException()
+IOException(String)

3 입력받기 전까지 무한 반복
잘못입력시 다시 입력받기

프로그램 실행 설명

잘못 입력하면 다시 입력받기

① Select schedule -> scheduleList 출력 & 번호 입력받기

무한반복

0입력시 처음으로

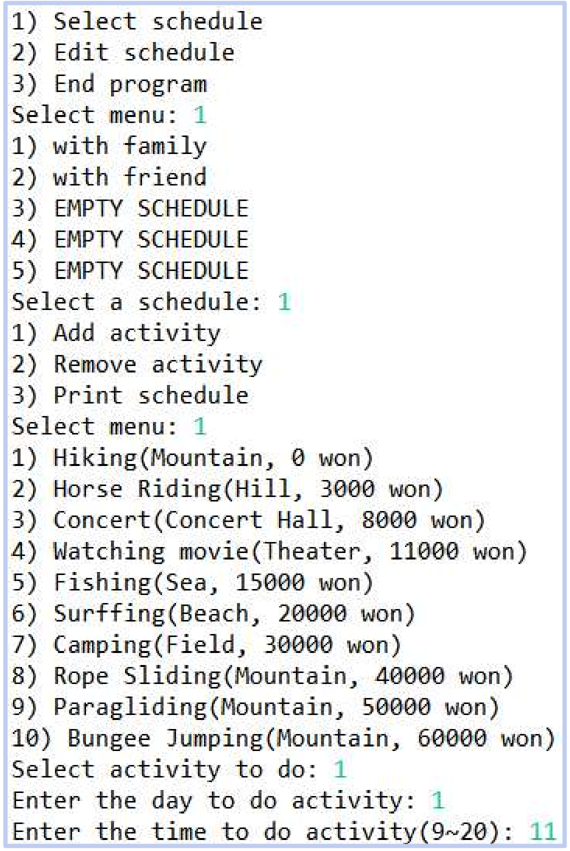
잘못입력시 다시입력받기

-> ① Add activity -> activityList 출력 & 번호, 날짜, 시간 입력받기

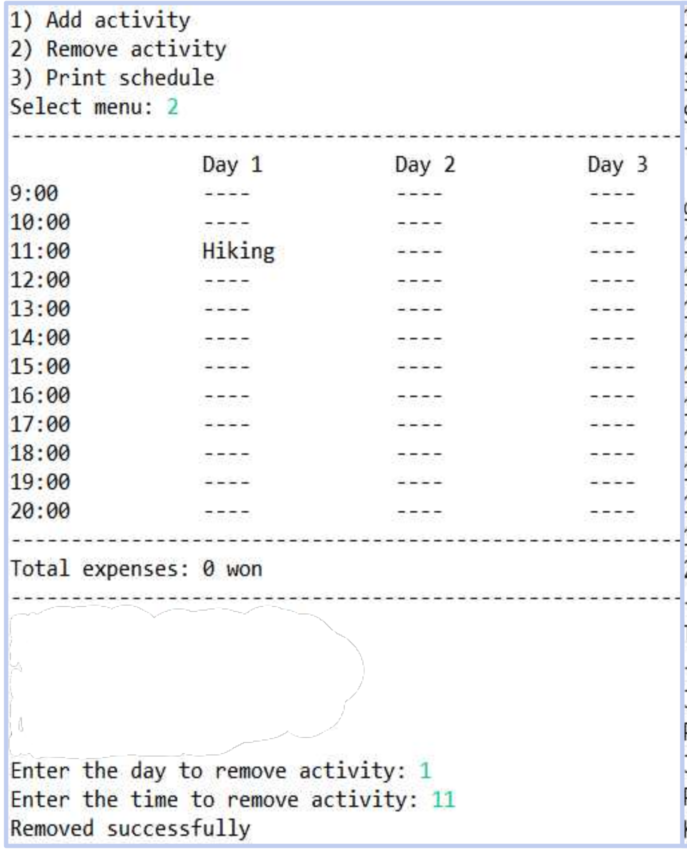
② Remove activity -> 전체 스케쥴표, 가격 출력 잘못입력하면 다시입력받기

&삭제할 날짜, 시간 입력받기

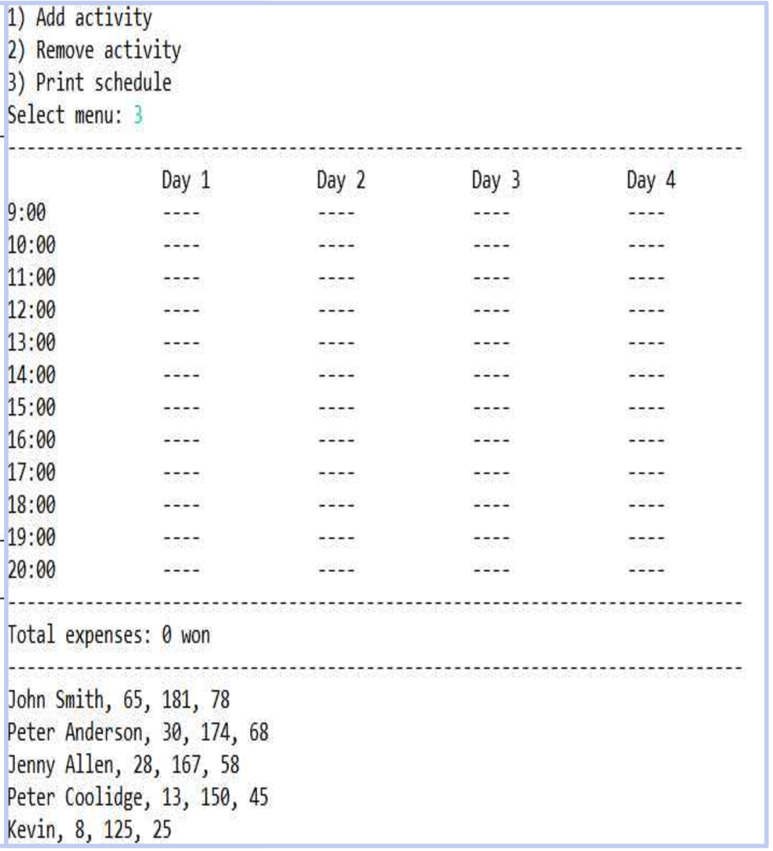
③ Print schedule -> 전체 스케쥴표, 가격, 멤버 출력



```
1) Select schedule
2) Edit schedule
3) End program
Select menu: 1
1) with family
2) with friend
3) EMPTY SCHEDULE
4) EMPTY SCHEDULE
5) EMPTY SCHEDULE
Select a schedule: 1
1) Add activity
2) Remove activity
3) Print schedule
Select menu: 1
1) Hiking(Mountain, 0 won)
2) Horse Riding(Hill, 3000 won)
3) Concert(Concert Hall, 8000 won)
4) Watching movie(Theater, 11000 won)
5) Fishing(Sea, 15000 won)
6) Surffing(Beach, 20000 won)
7) Camping(Field, 30000 won)
8) Rope Sliding(Mountain, 40000 won)
9) Paragliding(Mountain, 50000 won)
10) Bungee Jumping(Mountain, 60000 won)
Select activity to do: 1
Enter the day to do activity: 1
Enter the time to do activity(9~20): 11
```



```
1) Add activity
2) Remove activity
3) Print schedule
Select menu: 2
-----------------------------------------------------------------
                  Day 1              Day 2              Day 3
9:00              ----               ----               ----
10:00             ----               ----               ----
11:00             Hiking             ----               ----
12:00             ----               ----               ----
13:00             ----               ----               ----
14:00             ----               ----               ----
15:00             ----               ----               ----
16:00             ----               ----               ----
17:00             ----               ----               ----
18:00             ----               ----               ----
19:00             ----               ----               ----
20:00             ----               ----               ----
-----------------------------------------------------------------
Total expenses: 0 won
-----------------------------------------------------------------

Enter the day to remove activity: 1
Enter the time to remove activity: 11
Removed successfully
```



```
1) Add activity
2) Remove activity
3) Print schedule
Select menu: 3
-----------------------------------------------------------------
                  Day 1         Day 2         Day 3         Day 4
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
-----------------------------------------------------------------
John Smith, 65, 181, 78
Peter Anderson, 30, 174, 68
Jenny Allen, 28, 167, 58
Peter Coolidge, 13, 150, 45
Kevin, 8, 125, 25
```

② Edit schedule -> ① Make a new schedule -> 이름, days, 멤버 입력받아 sch 생성

잘못입력하면 다시 입력받기

무한반복

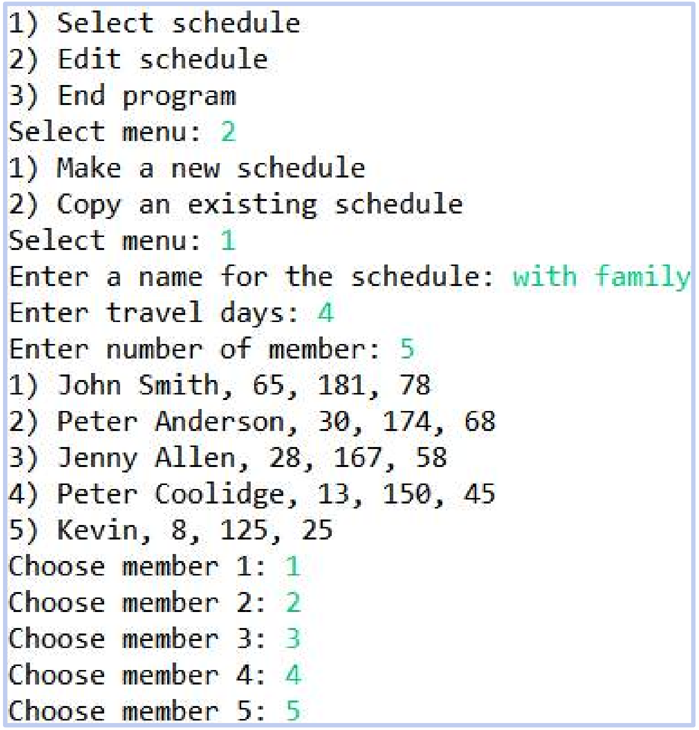
0 입력시 처음으로

② Copy an existing sch -> scheduleList 출력

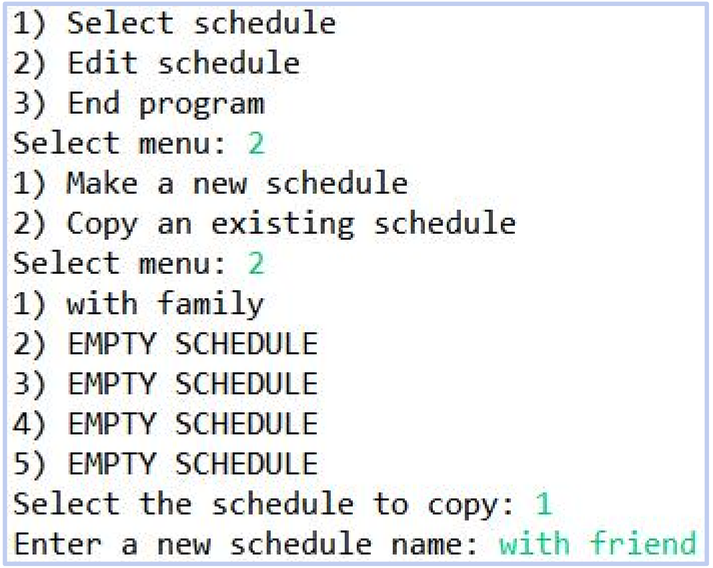
& 번호, 이름 입력받아 schedule 생성

잘못 입력하면 다시 입력받기

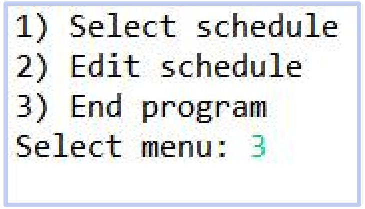
③ End program -> 프로그램 종료



```
1) Select schedule
2) Edit schedule
3) End program
Select menu: 2
1) Make a new schedule
2) Copy an existing schedule
Select menu: 1
Enter a name for the schedule: with family
Enter travel days: 4
Enter number of member: 5
1) John Smith, 65, 181, 78
2) Peter Anderson, 30, 174, 68
3) Jenny Allen, 28, 167, 58
4) Peter Coolidge, 13, 150, 45
5) Kevin, 8, 125, 25
Choose member 1: 1
Choose member 2: 2
Choose member 3: 3
Choose member 4: 4
Choose member 5: 5
```

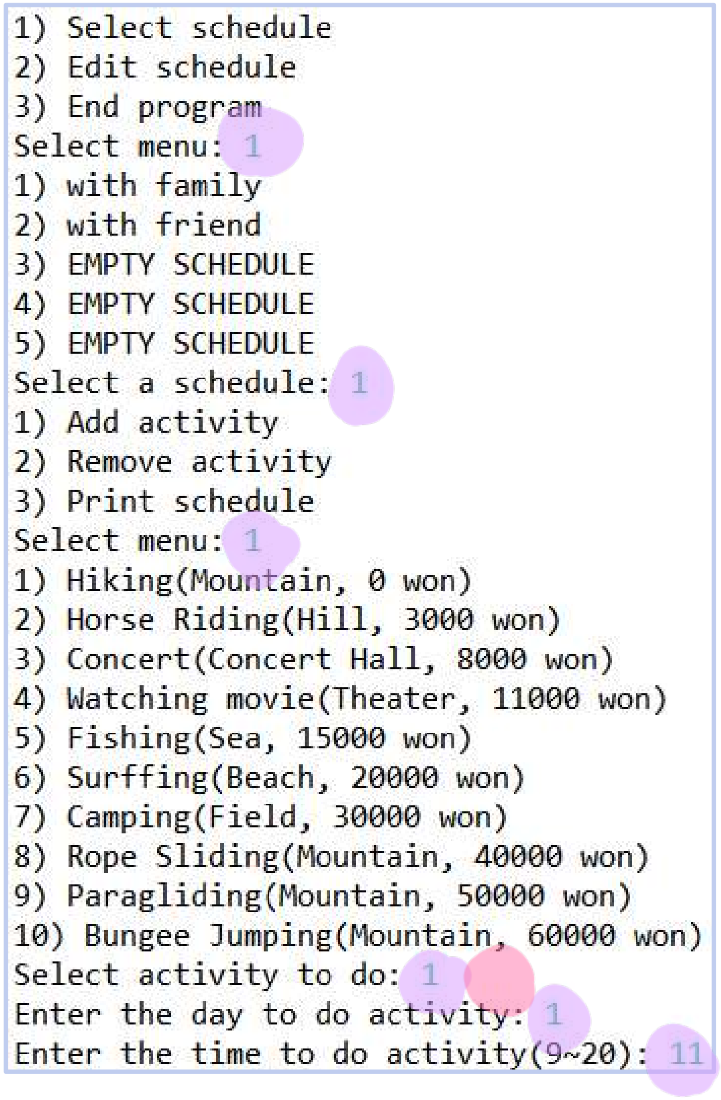


```
1) Select schedule
2) Edit schedule
3) End program
Select menu: 2
1) Make a new schedule
2) Copy an existing schedule
Select menu: 2
1) with family
2) EMPTY SCHEDULE
3) EMPTY SCHEDULE
4) EMPTY SCHEDULE
5) EMPTY SCHEDULE
Select the schedule to copy: 1
Enter a new schedule name: with friend
```

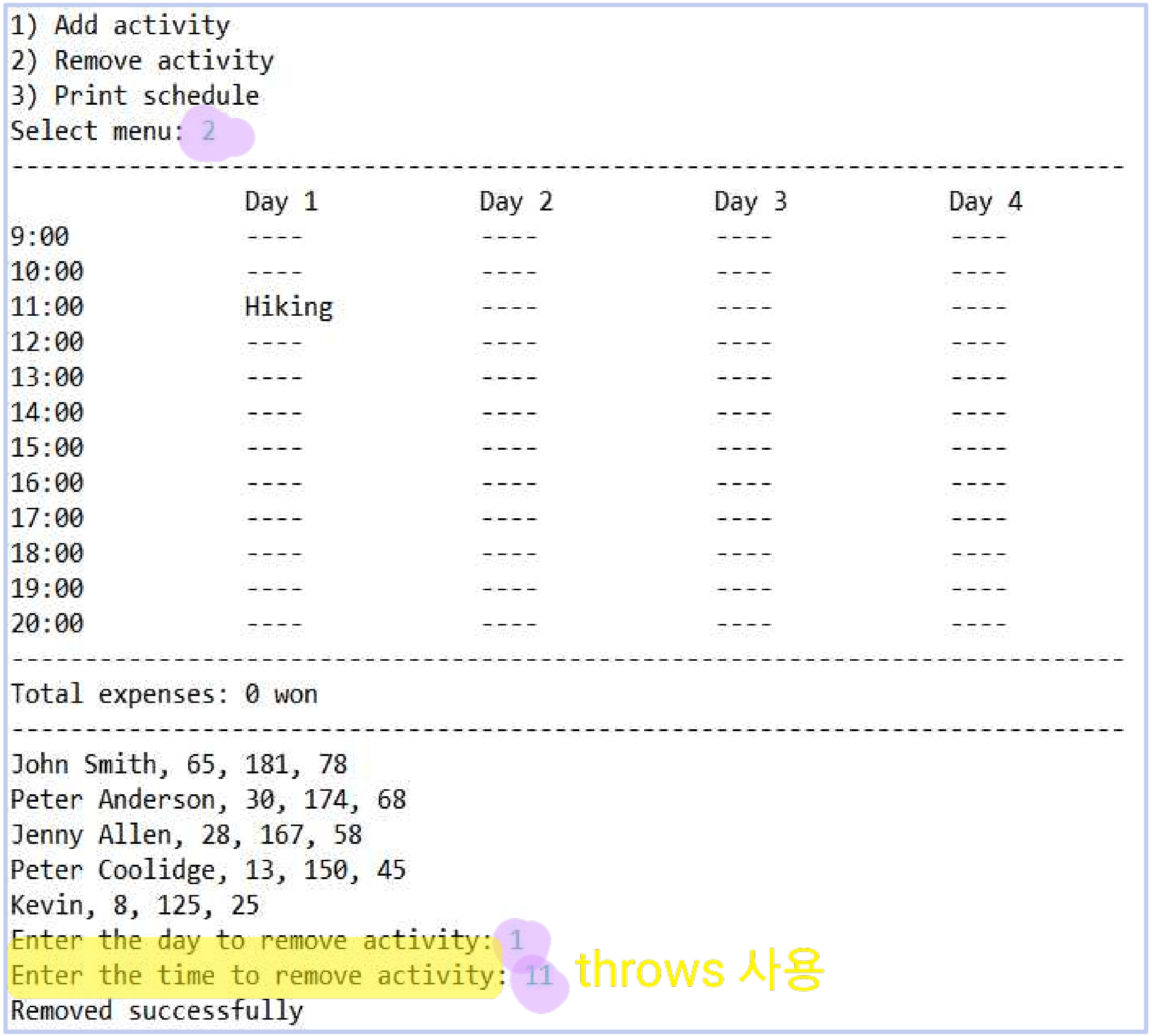


```
1) Select schedule
2) Edit schedule
3) End program
Select menu: 3
```

Exception Handling



```
1) Select schedule
2) Edit schedule
3) End program
Select menu: 1
1) with family
2) with friend
3) EMPTY SCHEDULE
4) EMPTY SCHEDULE
5) EMPTY SCHEDULE
Select a schedule: 1
1) Add activity
2) Remove activity
3) Print schedule
Select menu: 1
1) Hiking(Mountain, 0 won)
2) Horse Riding(Hill, 3000 won)
3) Concert(Concert Hall, 8000 won)
4) Watching movie(Theater, 11000 won)
5) Fishing(Sea, 15000 won)
6) Surffing(Beach, 20000 won)
7) Camping(Field, 30000 won)
8) Rope Sliding(Mountain, 40000 won)
9) Paragliding(Mountain, 50000 won)
10) Bungee Jumping(Mountain, 60000 won)
Select activity to do: 1
Enter the day to do activity: 1
Enter the time to do activity(9~20): 11
```



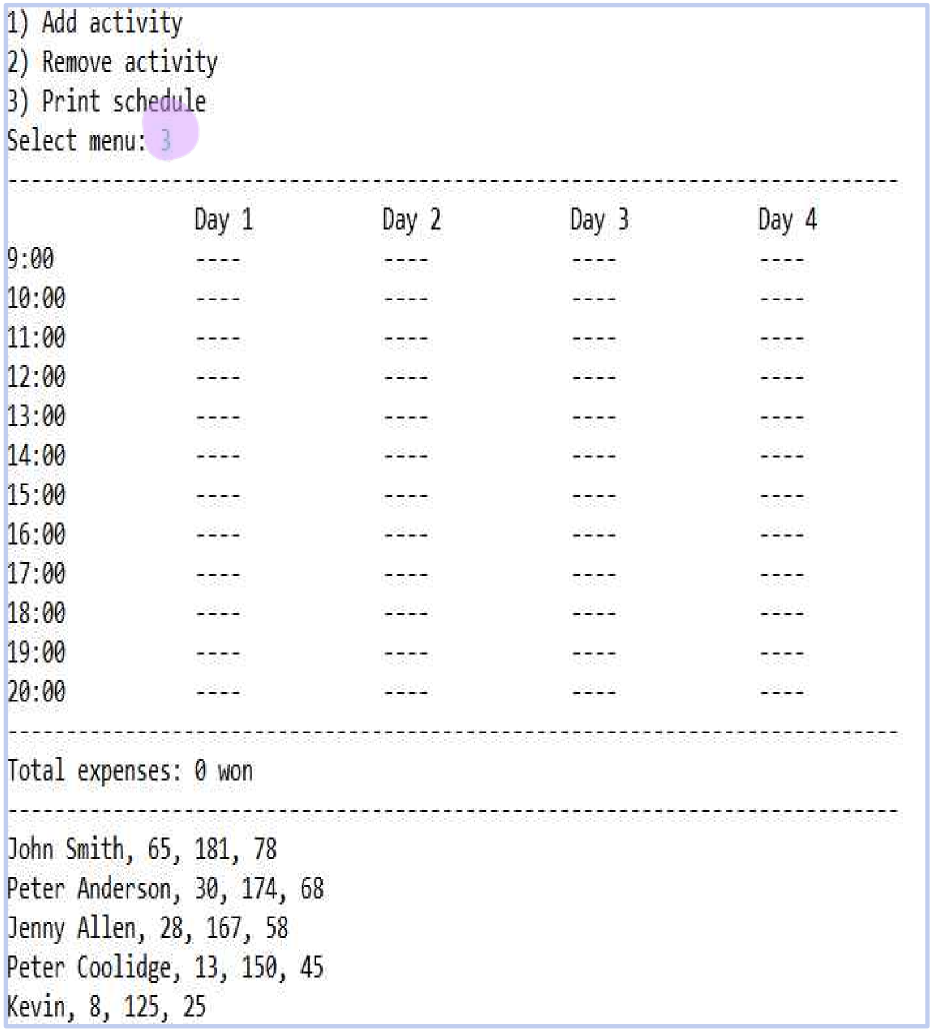
```
1) Add activity
2) Remove activity
3) Print schedule
Select menu: 2
-----------------------------------------------------------------------------
                 Day 1            Day 2            Day 3            Day 4
9:00             ----             ----             ----             ----
10:00            ----             ----             ----             ----
11:00            Hiking           ----             ----             ----
12:00            ----             ----             ----             ----
13:00            ----             ----             ----             ----
14:00            ----             ----             ----             ----
15:00            ----             ----             ----             ----
16:00            ----             ----             ----             ----
17:00            ----             ----             ----             ----
18:00            ----             ----             ----             ----
19:00            ----             ----             ----             ----
20:00            ----             ----             ----             ----
-----------------------------------------------------------------------------
Total expenses: 0 won
-----------------------------------------------------------------------------
John Smith, 65, 181, 78
Peter Anderson, 30, 174, 68
Jenny Allen, 28, 167, 58
Peter Coolidge, 13, 150, 45
Kevin, 8, 125, 25
Enter the day to remove activity: 1
Enter the time to remove activity: 11
Removed successfully
```

throws 사용

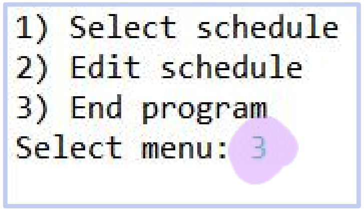
InvalidAccessException

ArrayFullException

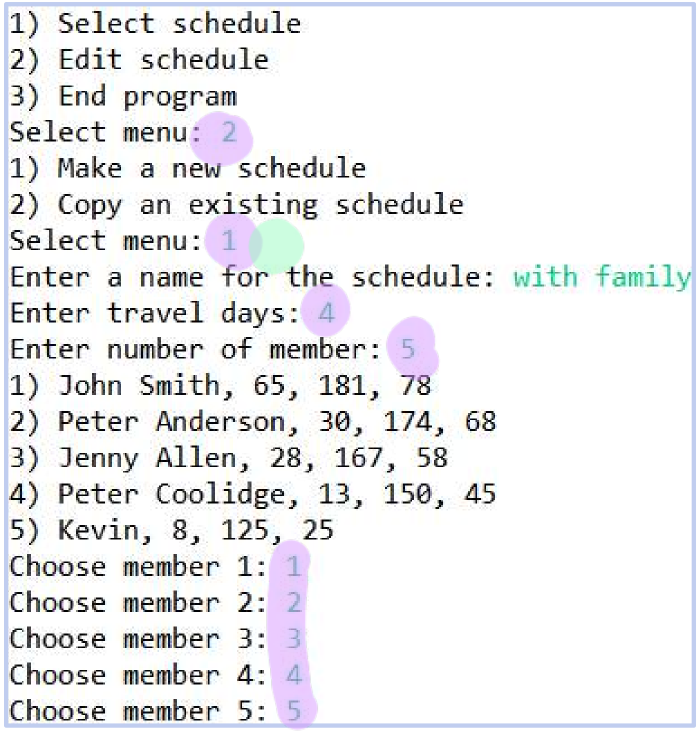
InsufficientConditionException



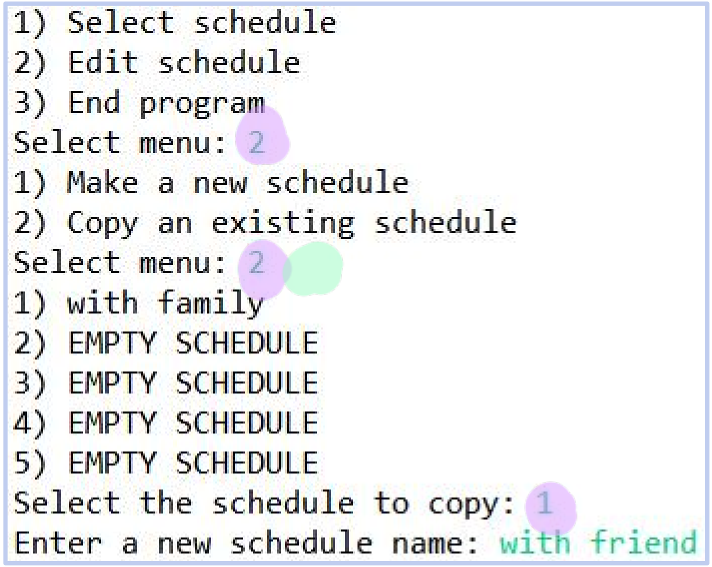
```
1) Add activity
2) Remove activity
3) Print schedule
Select menu: 3
------------------------------------------------------------------------------
                Day 1           Day 2           Day 3           Day 4
9:00            ----            ----            ----            ----
10:00           ----            ----            ----            ----
11:00           ----            ----            ----            ----
12:00           ----            ----            ----            ----
13:00           ----            ----            ----            ----
14:00           ----            ----            ----            ----
15:00           ----            ----            ----            ----
16:00           ----            ----            ----            ----
17:00           ----            ----            ----            ----
18:00           ----            ----            ----            ----
19:00           ----            ----            ----            ----
20:00           ----            ----            ----            ----
------------------------------------------------------------------------------
Total expenses: 0 won
------------------------------------------------------------------------------
John Smith, 65, 181, 78
Peter Anderson, 30, 174, 68
Jenny Allen, 28, 167, 58
Peter Coolidge, 13, 150, 45
Kevin, 8, 125, 25
```



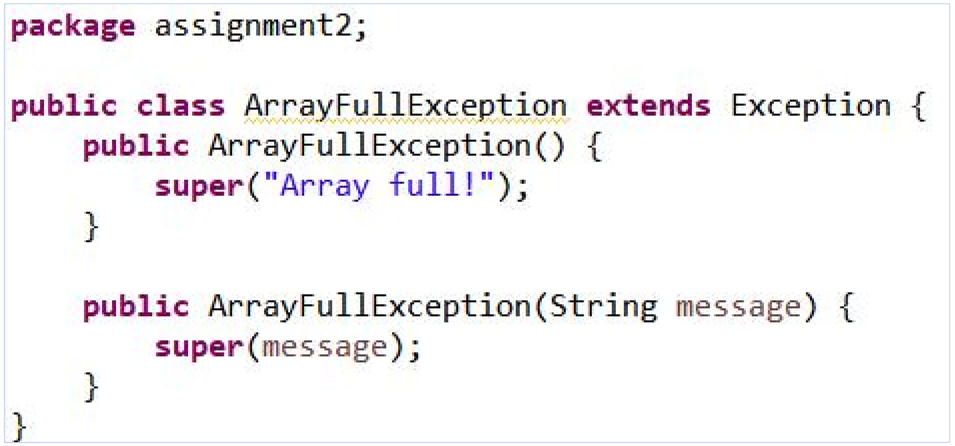
```
1) Select schedule
2) Edit schedule
3) End program
Select menu: 3
```



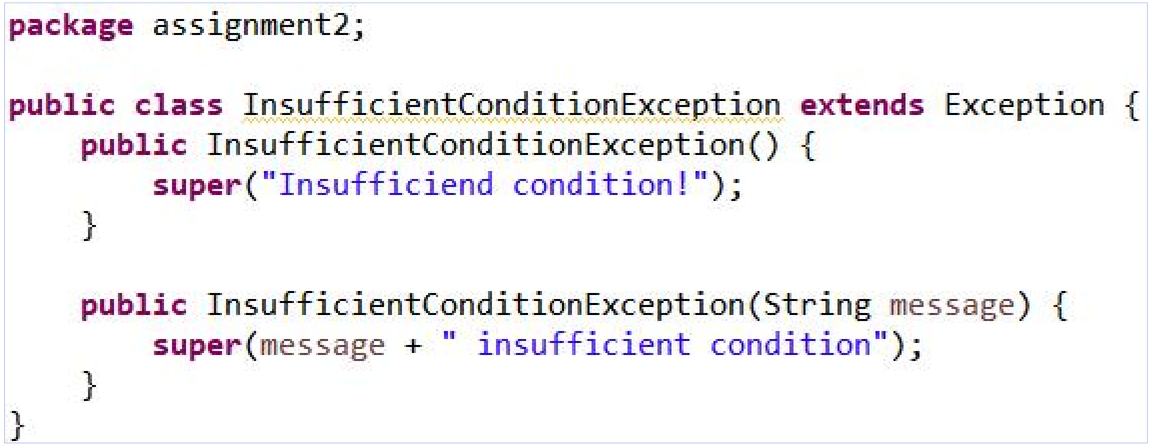
```
1) Select schedule
2) Edit schedule
3) End program
Select menu: 2
1) Make a new schedule
2) Copy an existing schedule
Select menu: 1
Enter a name for the schedule: with family
Enter travel days: 4
Enter number of member: 5
1) John Smith, 65, 181, 78
2) Peter Anderson, 30, 174, 68
3) Jenny Allen, 28, 167, 58
4) Peter Coolidge, 13, 150, 45
5) Kevin, 8, 125, 25
Choose member 1: 1
Choose member 2: 2
Choose member 3: 3
Choose member 4: 4
Choose member 5: 5
```



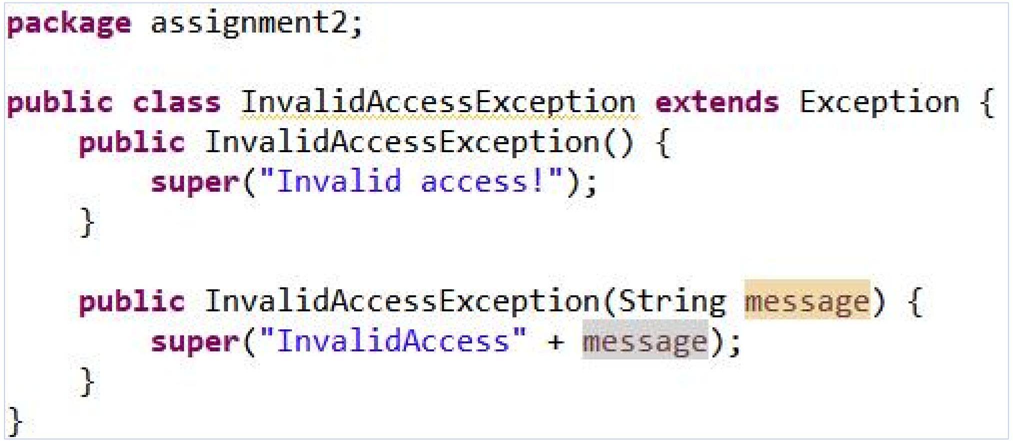
```
1) Select schedule
2) Edit schedule
3) End program
Select menu: 2
1) Make a new schedule
2) Copy an existing schedule
Select menu: 2
1) with family
2) EMPTY SCHEDULE
3) EMPTY SCHEDULE
4) EMPTY SCHEDULE
5) EMPTY SCHEDULE
Select the schedule to copy: 1
Enter a new schedule name: with friend
```



```java
package assignment2;

public class ArrayFullException extends Exception {
    public ArrayFullException() {
        super("Array full!");
    }

    public ArrayFullException(String message) {
        super(message);
    }
}
```



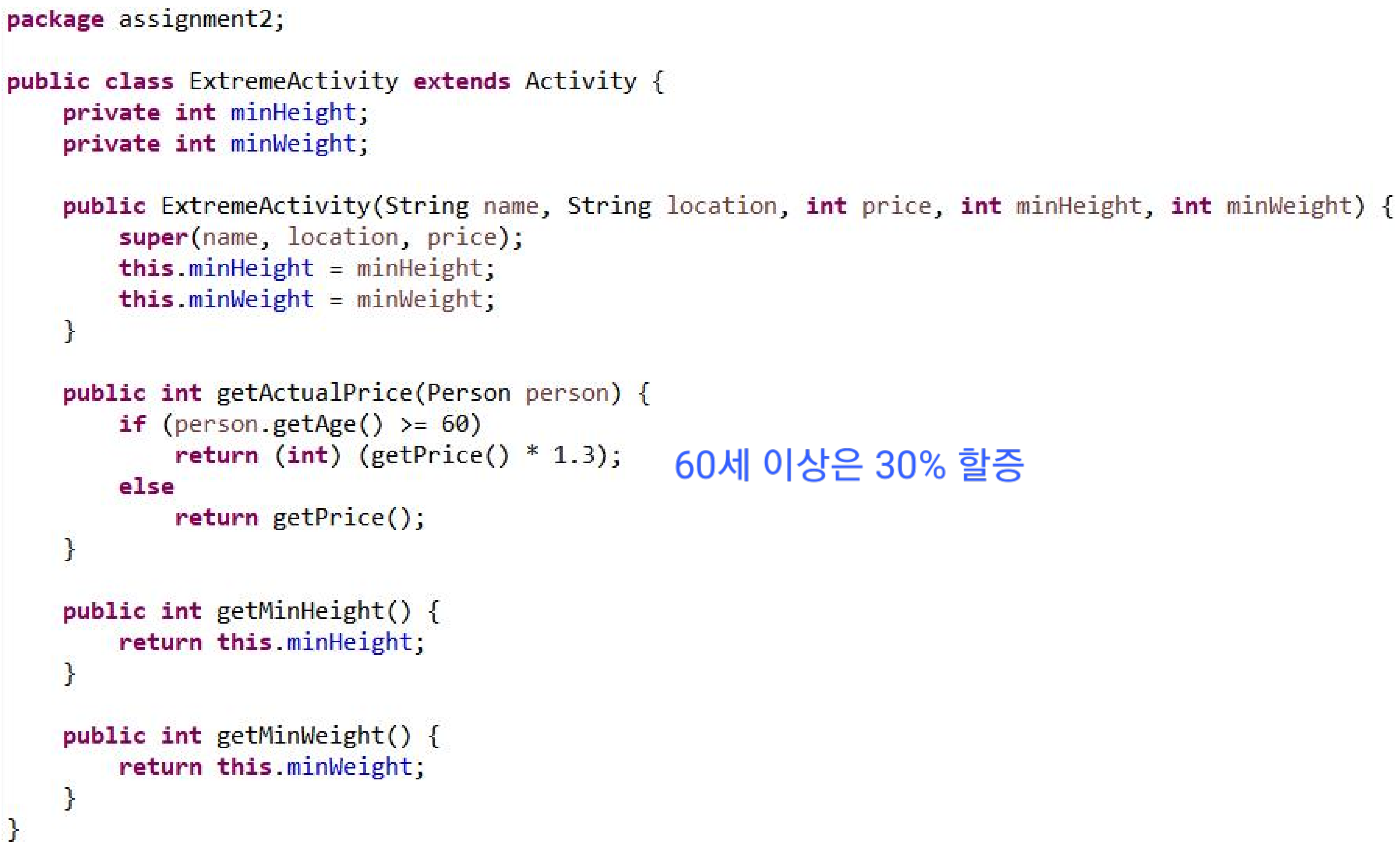
```java
package assignment2;

public class InsufficientConditionException extends Exception {
    public InsufficientConditionException() {
        super("Insufficiend condition!");
    }

    public InsufficientConditionException(String message) {
        super(message + " insufficient condition");
    }
}
```



```java
package assignment2;

public class InvalidAccessException extends Exception {
    public InvalidAccessException() {
        super("Invalid access!");
    }

    public InvalidAccessException(String message) {
        super("InvalidAccess" + message);
    }
}
```

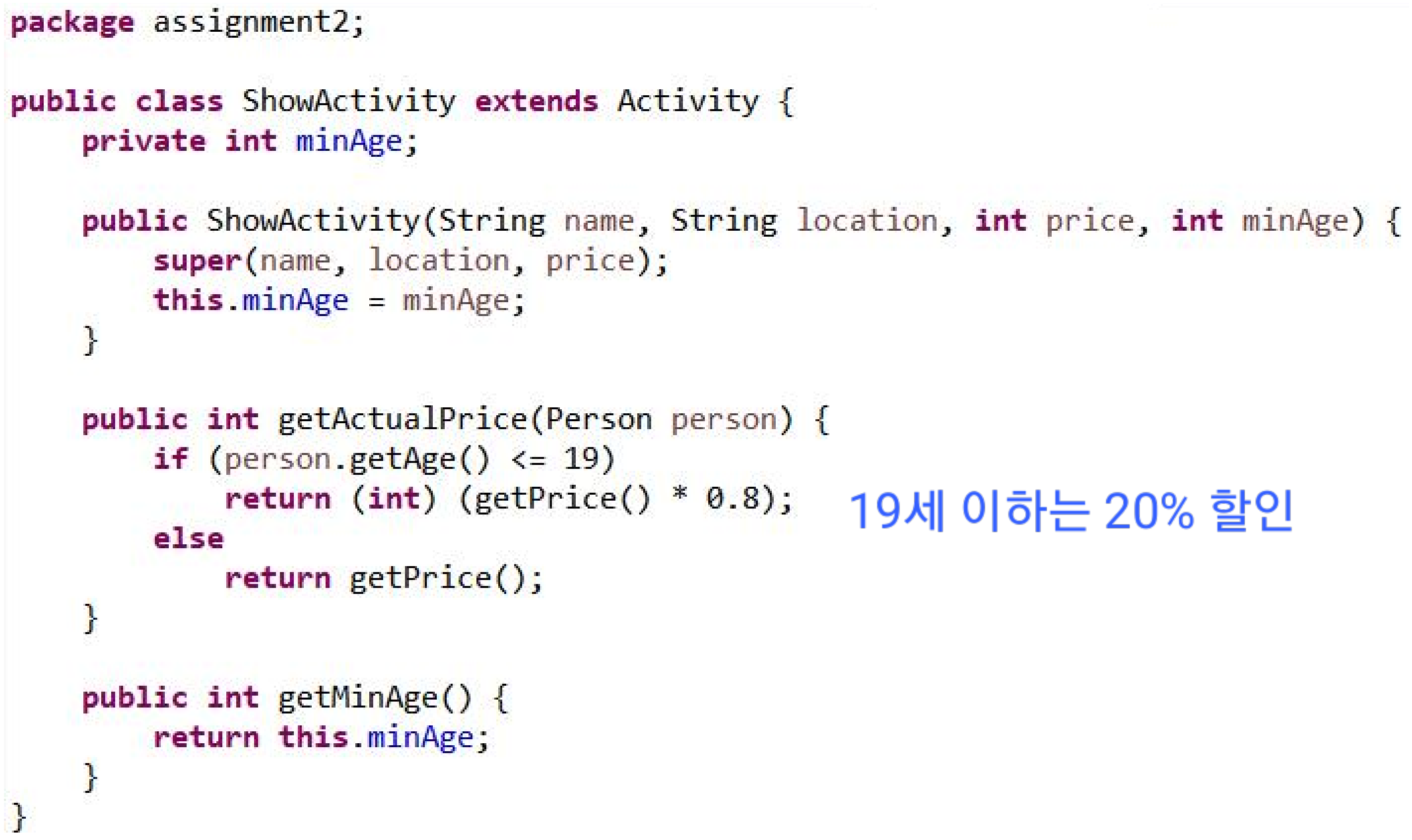
```java
package assignment2;

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() {
        return name + "(" + location + ", " + price + " won)";
    }

    public int getPrice() {
        return this.price;
    }

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

    public int getActualPrice(Person person) {
        return price;
    }

    public boolean eqauls(Object obj) {
        if (obj == null)
            return false;
        else if (getClass() != obj.getClass())
            return false;
        else {
            Activity a = (Activity) obj;
            return name.equals(a.name) && location.equals(a.location) && price == a.price;
        }
    }
}
```

이미 존재하는 activity를 추가하려할 때 사용



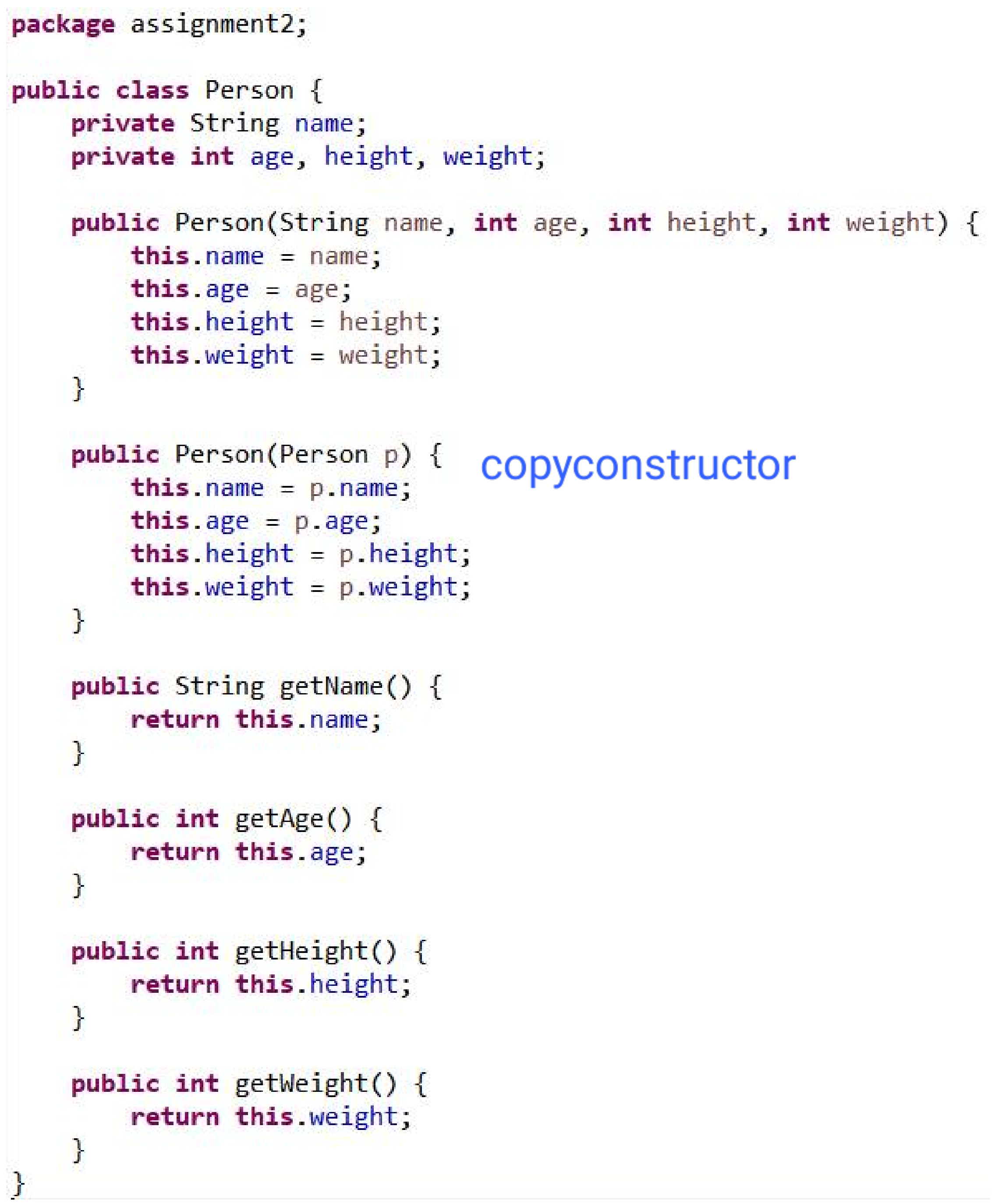
```java
package assignment2;

public class ExtremeActivity extends Activity {
    private int minHeight;
    private int minWeight;

    public ExtremeActivity(String name, String location, int price, int minHeight, int minWeight) {
        super(name, location, price);
        this.minHeight = minHeight;
        this.minWeight = minWeight;
    }

    public int getActualPrice(Person person) {
        if (person.getAge() >= 60)
            return (int) (getPrice() * 1.3);
        else
            return getPrice();
    }

    public int getMinHeight() {
        return this.minHeight;
    }

    public int getMinWeight() {
        return this.minWeight;
    }
}
```

60세 이상은 30% 할증



```java
package assignment2;

public class ShowActivity extends Activity {
    private int minAge;

    public ShowActivity(String name, String location, int price, int minAge) {
        super(name, location, price);
        this.minAge = minAge;
    }

    public int getActualPrice(Person person) {
        if (person.getAge() <= 19)
            return (int) (getPrice() * 0.8);
        else
            return getPrice();
    }

    public int getMinAge() {
        return this.minAge;
    }
}
```

19세 이하는 20% 할인

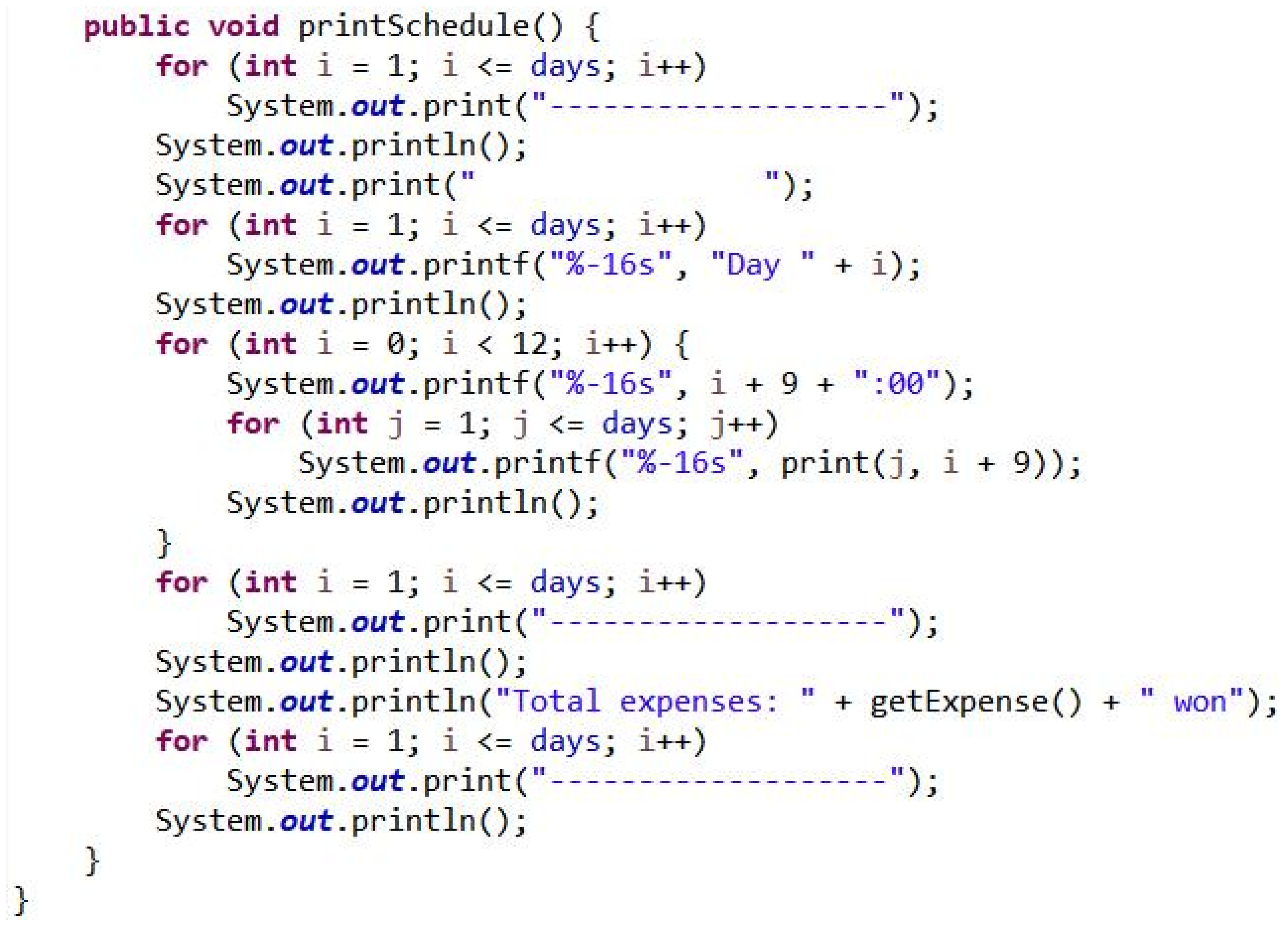


```java
package assignment2;

public class Person {
    private String name;
    private int age, height, weight;

    public Person(String name, int age, int height, int weight) {
        this.name = name;
        this.age = age;
        this.height = height;
        this.weight = weight;
    }

    public Person(Person p) {
        this.name = p.name;
        this.age = p.age;
        this.height = p.height;
        this.weight = p.weight;
    }

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

    public int getAge() {
        return this.age;
    }

    public int getHeight() {
        return this.height;
    }

    public int getWeight() {
        return this.weight;
    }
}
```

copyconstructor

```java
package assignment2;

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

    public Schedule(String name, int days, Person[] member) {
        this.name = name;
        this.days = days;
        this.expense = 0;
        this.plan = new Activity[days][12];
        this.member = new Person[member.length];
        for (int i = 0; i < member.length; i++)
            this.member[i] = new Person(member[i]);
        scheduleNum++;
```
copyconstructor 이용
```java
    }

    public Schedule(String name, Schedule s1, Person[] member) {
        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];
        this.member = new Person[member.length];
        for (int i = 0; i < member.length; i++)
            this.member[i] = new Person(member[i]);
        scheduleNum++;
```
copyconstructor 이용
```java
    }

    public int getDays() {
        return this.days;
    }

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

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

public Person[] getMember() {
    return this.member;
}

public int setPlan(Activity activity, int day, int time) {
    if (plan[day - 1][time - 9] != null)
        return 0;
    for (int i = 0; i < this.days; i++)
        for (int j = 0; j < 12; j++)
            if (activity.equals(this.plan[i][j]))
                return 0;

    this.plan[day - 1][time - 9] = activity;
    for (Person m : member)
        this.expense += activity.getActualPrice(m);
    return 1;
```
activity의 type에 따라 late binding
```java
}

public int removePlan(int day, int time) throws InvalidAccessException {
```
throws 사용
```java
    if (plan[day - 1][time - 9] == null)
        throw new InvalidAccessException();
    for (Person m : member)
        this.expense -= this.plan[day - 1][time - 9].getActualPrice(m);
    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();
}
```

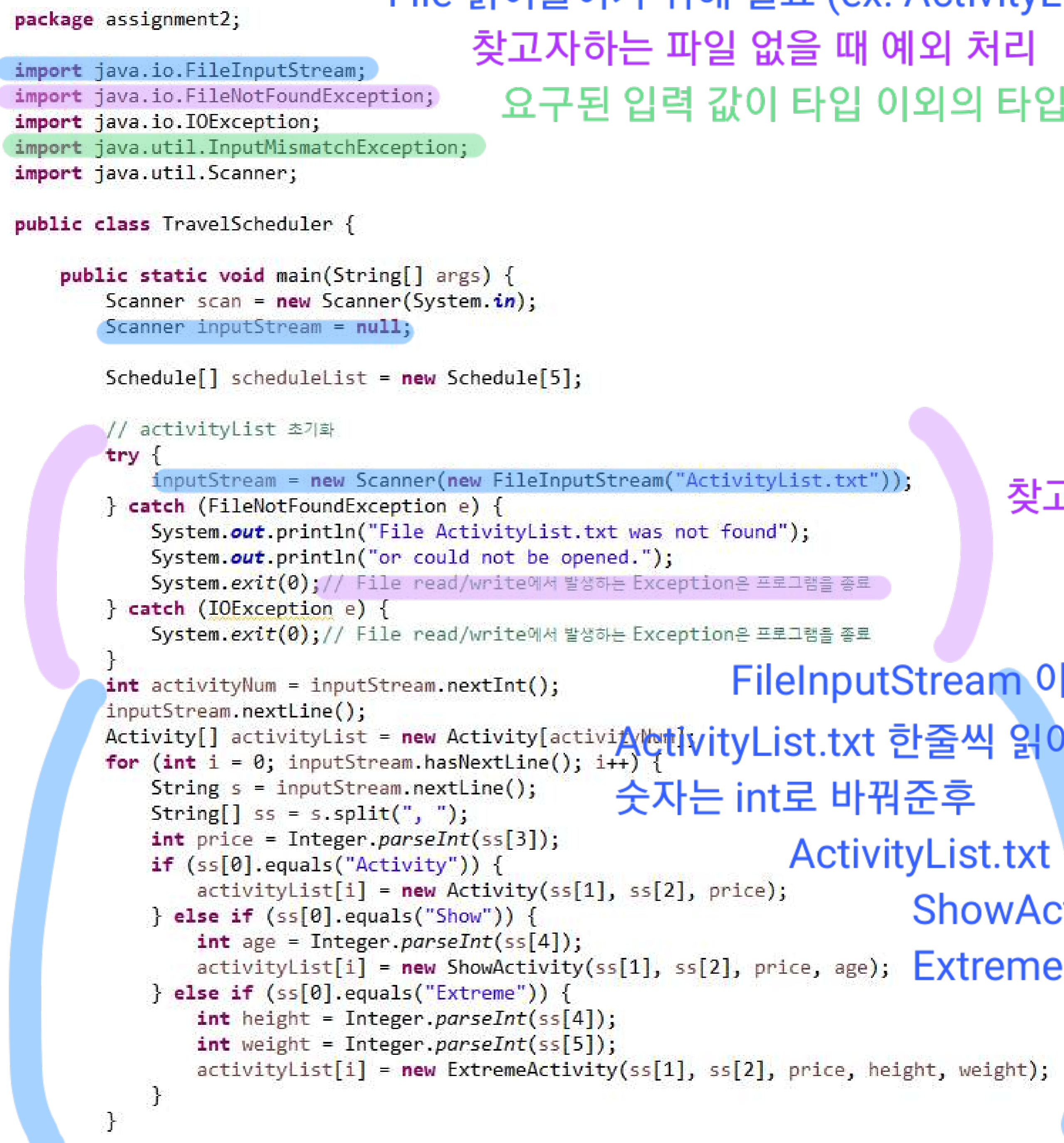


```java
    public void printSchedule() {
        for (int i = 1; i <= days; i++)
            System.out.print("-------------------");
        System.out.println();
        System.out.print("                ");
        for (int i = 1; i <= days; i++)
            System.out.printf("%-16s", "Day " + i);
        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();
    }
}
```

File 읽어들이기 위해 필요 (ex. ActivityList.txt, MemberList.txt)

찾고자하는 파일 없을 때 예외 처리

요구된 입력 값이 타입 이외의 타입을 입력 받을때 예외 처리

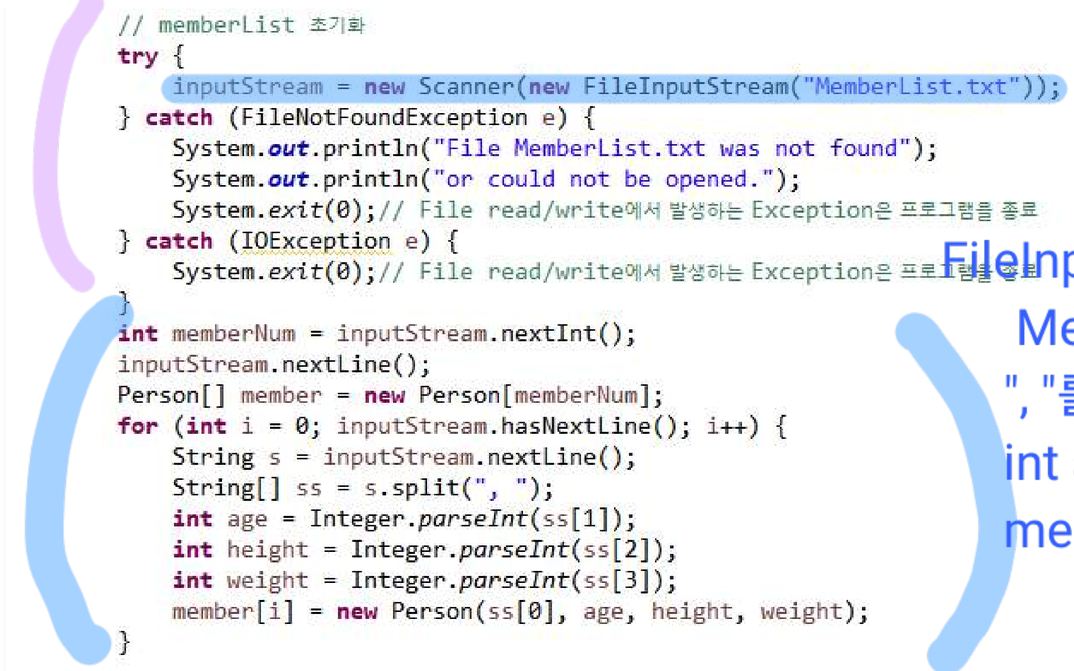


```java
package assignment2;

import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.InputMismatchException;
import java.util.Scanner;

public class TravelScheduler {

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

        Schedule[] scheduleList = new Schedule[5];

        // activityList 초기화
        try {
            inputStream = new Scanner(new FileInputStream("ActivityList.txt"));
        } catch (FileNotFoundException e) {
            System.out.println("File ActivityList.txt was not found");
            System.out.println("or could not be opened.");
            System.exit(0);// File read/write에서 발생하는 Exception은 프로그램을 종료
        } catch (IOException e) {
            System.exit(0);// File read/write에서 발생하는 Exception은 프로그램을 종료
        }
        int activityNum = inputStream.nextInt();
        inputStream.nextLine();
        Activity[] activityList = new Activity[activityNum];
        for (int i = 0; inputStream.hasNextLine(); i++) {
            String s = inputStream.nextLine();
            String[] ss = s.split(", ");
            int price = Integer.parseInt(ss[3]);
            if (ss[0].equals("Activity")) {
                activityList[i] = new Activity(ss[1], ss[2], price);
            } else if (ss[0].equals("Show")) {
                int age = Integer.parseInt(ss[4]);
                activityList[i] = new ShowActivity(ss[1], ss[2], price, age);
            } else if (ss[0].equals("Extreme")) {
                int height = Integer.parseInt(ss[4]);
                int weight = Integer.parseInt(ss[5]);
                activityList[i] = new ExtremeActivity(ss[1], ss[2], price, height, weight);
            }
        }
```

찾고자하는 파일 없을 때 예외 처리

FileInputStream 이용해
ActivityList.txt 한줄씩 읽어들인 후 ", "를 기준으로 String 나눠주고
숫자는 int로 바꿔준후

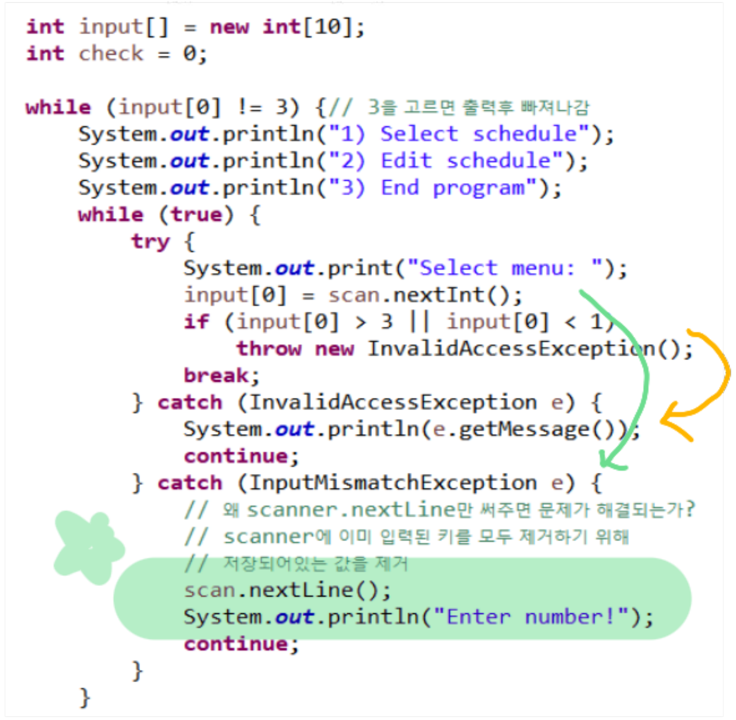
ActivityList.txt 각 줄 맨앞의 Activity type 별로 Activity,
ShowActivity,
ExtremeActivity 객체 만들어줌



```java
// memberList 초기화
try {
    inputStream = new Scanner(new FileInputStream("MemberList.txt"));
} catch (FileNotFoundException e) {
    System.out.println("File MemberList.txt was not found");
    System.out.println("or could not be opened.");
    System.exit(0);// File read/write에서 발생하는 Exception은 프로그램을 종료
} catch (IOException e) {
    System.exit(0);// File read/write에서 발생하는 Exception은 프로그램을 종료
}
int memberNum = inputStream.nextInt();
inputStream.nextLine();
Person[] member = new Person[memberNum];
for (int i = 0; inputStream.hasNextLine(); i++) {
    String s = inputStream.nextLine();
    String[] ss = s.split(", ");
    int age = Integer.parseInt(ss[1]);
    int height = Integer.parseInt(ss[2]);
    int weight = Integer.parseInt(ss[3]);
    member[i] = new Person(ss[0], age, height, weight);
}
```

찾고자하는 파일 없을 때 예외 처리

FileInputStream 이용해
MemberList.txt 한줄씩 읽어들인 후
", "를 기준으로 String 나눠
int age, height, weight로 만들어
memberList 요소의 객체 만들기

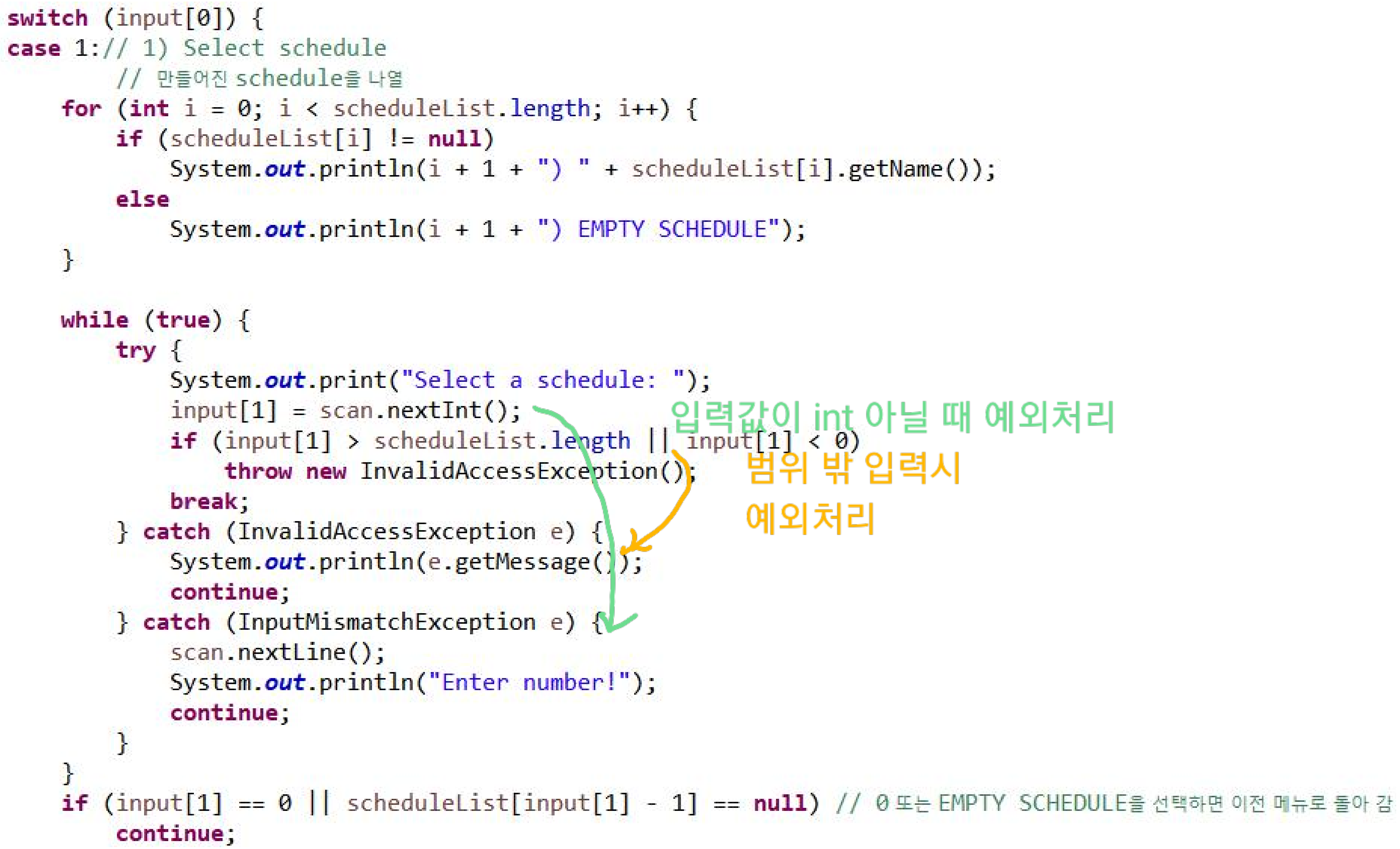


```java
int input[] = new int[10];
int check = 0;

while (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");
    while (true) {
        try {
            System.out.print("Select menu: ");
            input[0] = scan.nextInt();
            if (input[0] > 3 || input[0] < 1)
                throw new InvalidAccessException();
            break;
        } catch (InvalidAccessException e) {
            System.out.println(e.getMessage());
            continue;
        } catch (InputMismatchException e) {
            // 왜 scanner.nextLine만 써주면 문제가 해결되는가?
            // scanner에 이미 입력된 키를 모두 제거하기 위해
            // 저장되어있는 값을 제거
            scan.nextLine();
            System.out.println("Enter number!");
            continue;
        }
    }
```

입력값이 int 아닐 때 예외처리

범위 밖 입력시
예외처리

scan.nextLine() 안해주면 앞 내용 무한반복출력됨



```java
switch (input[0]) {
case 1:// 1) Select schedule
        // 만들어진 schedule을 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");
    }

    while (true) {
        try {
            System.out.print("Select a schedule: ");
            input[1] = scan.nextInt();
            if (input[1] > scheduleList.length || input[1] < 0)
                throw new InvalidAccessException();
            break;
        } catch (InvalidAccessException e) {
            System.out.println(e.getMessage());
            continue;
        } catch (InputMismatchException e) {
            scan.nextLine();
            System.out.println("Enter number!");
            continue;
        }
    }
    if (input[1] == 0 || scheduleList[input[1] - 1] == null) // 0 또는 EMPTY SCHEDULE을 선택하면 이전 메뉴로 돌아 감
        continue;
```

입력값이 int 아닐 때 예외처리

범위 밖 입력시
예외처리

아래를 실행하지 않고 이전 메뉴로 돌아가기

올바르게
입력할때까지
무한반복

```java
do {
    System.out.println("1) Add activity");
    System.out.println("2) Remove activity");
    System.out.println("3) Print schedule");
    while (true) {
        System.out.print("Select menu: ");
        try {
            input[2] = scan.nextInt();
            if (input[2] > 3 || input[2] < 0)
                throw new InvalidAccessException();
            break;
        } catch (InvalidAccessException e) {
            System.out.println(e.getMessage());
            continue;
        } catch (InputMismatchException e) {
            scan.nextLine();
            System.out.println("Enter number!");
            continue;
        }
    }

    switch (input[2]) {
    case 0:
        break;
    case 1:// input[1]에 add activity
        int occur = 0;
        do {
            occur = 0;
            for (int i = 0; i < activityList.length; i++)
                System.out.println(i + 1 + ") " + activityList[i].toString());

            while (true) {
                try {
                    System.out.print("Select activity to do: ");
                    input[3] = scan.nextInt();
                    if (input[3] > activityList.length || input[3] < 1)
                        throw new InvalidAccessException("Activity");
                    break;
                } catch (InvalidAccessException e) {
                    System.out.println(e.getMessage());
                    continue;
                } catch (InputMismatchException e) {
                    scan.nextLine();
                    System.out.println("Enter number!");
                    continue;
                }
            }
```

```java
switch (input[2]) {
case 0:
    break;
case 1:// input[1]에 add activity
    int occur = 0;
    do {
        occur = 0;
        for (int i = 0; i < activityList.length; i++)
            System.out.println(i + 1 + ") " + activityList[i].toString());

        while (true) {
            try {
                System.out.print("Select activity to do: ");
                input[3] = scan.nextInt();
                if (input[3] > activityList.length || input[3] < 1)
                    throw new InvalidAccessException("Activity");
                break;
            } catch (InvalidAccessException e) {
                System.out.println(e.getMessage());
                continue;
            } catch (InputMismatchException e) {
                scan.nextLine();
                System.out.println("Enter number!");
                continue;
            }
        }

        while (true) {
            try {
                System.out.print("Enter the day to do activity: ");
                input[4] = scan.nextInt();
                if (input[4] > scheduleList[input[1] - 1].getDays() || input[4] < 1)
                    throw new InvalidAccessException(" Day");
                break;
            } catch (InvalidAccessException e) {
                System.out.println(e.getMessage());
                continue;
            } catch (InputMismatchException e) {
                scan.nextLine();
                System.out.println("Enter number!");
                continue;
            }
        while (true) {
            try {
                System.out.print("Enter the time to do activity(9~20): ");
                input[5] = scan.nextInt();
                if (input[5] > 20 || input[5] < 9)
                    throw new InvalidAccessException(" Time");
                break;
            } catch (InvalidAccessException e) {
                System.out.println(e.getMessage());
                continue;
            } catch (InputMismatchException e) {
                scan.nextLine();
                System.out.println("Enter number!");
                continue;
            }
        }

        try {
            if (activityList[input[3] - 1] instanceof ShowActivity) {
                ShowActivity s = (ShowActivity) activityList[input[3] - 1];
                for (Person m : scheduleList[input[1] - 1].getMember())
                    if (m.getAge() < s.getMinAge())
                        throw new InsufficientConditionException("age");
                check = scheduleList[input[1] - 1].setPlan(s, input[4], input[5]);
            } else if (activityList[input[3] - 1] instanceof ExtremeActivity) {
                ExtremeActivity e = (ExtremeActivity) activityList[input[3] - 1];
                for (Person m : scheduleList[input[1] - 1].getMember())
                    if (m.getHeight() < e.getMinHeight() || m.getWeight() < e.getMinWeight())
                        throw new InsufficientConditionException("height of weight");
                check = scheduleList[input[1] - 1].setPlan(e, input[4], input[5]);
            } else
                check = scheduleList[input[1] - 1].setPlan(activityList[input[3] - 1], input[4],
                        input[5]);

            if (check == 0)
                System.out.println("Fail to add activity");
        } catch (InsufficientConditionException e) {
            System.out.println(e.getMessage());
            occur = 1;
        }
    } while (occur == 1);
    break;
```

입력받은 Activity가 'Show' 일때
Activity 객체를 ShowActivity로 downcasting

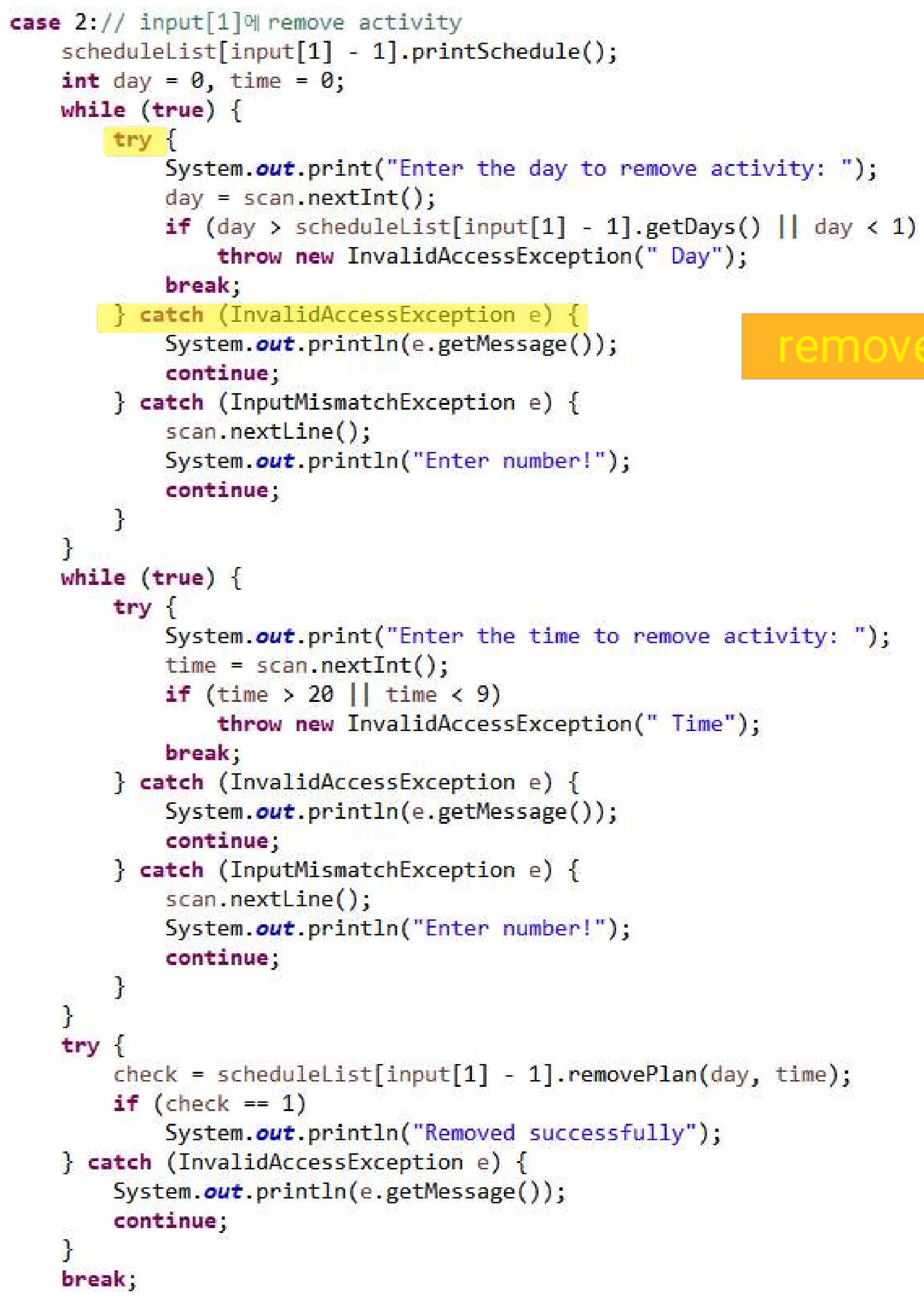
멤버의 나이가 activity의 최소 연령보다
작을 때 예외 처리

s의 type은 ShowActivity이므로 showActivity의
getActualPrice()와 binding,

입력받은 Activity가 'Extreme' 일 때
Activity 객체를 ExtremeActivity로 downcasting

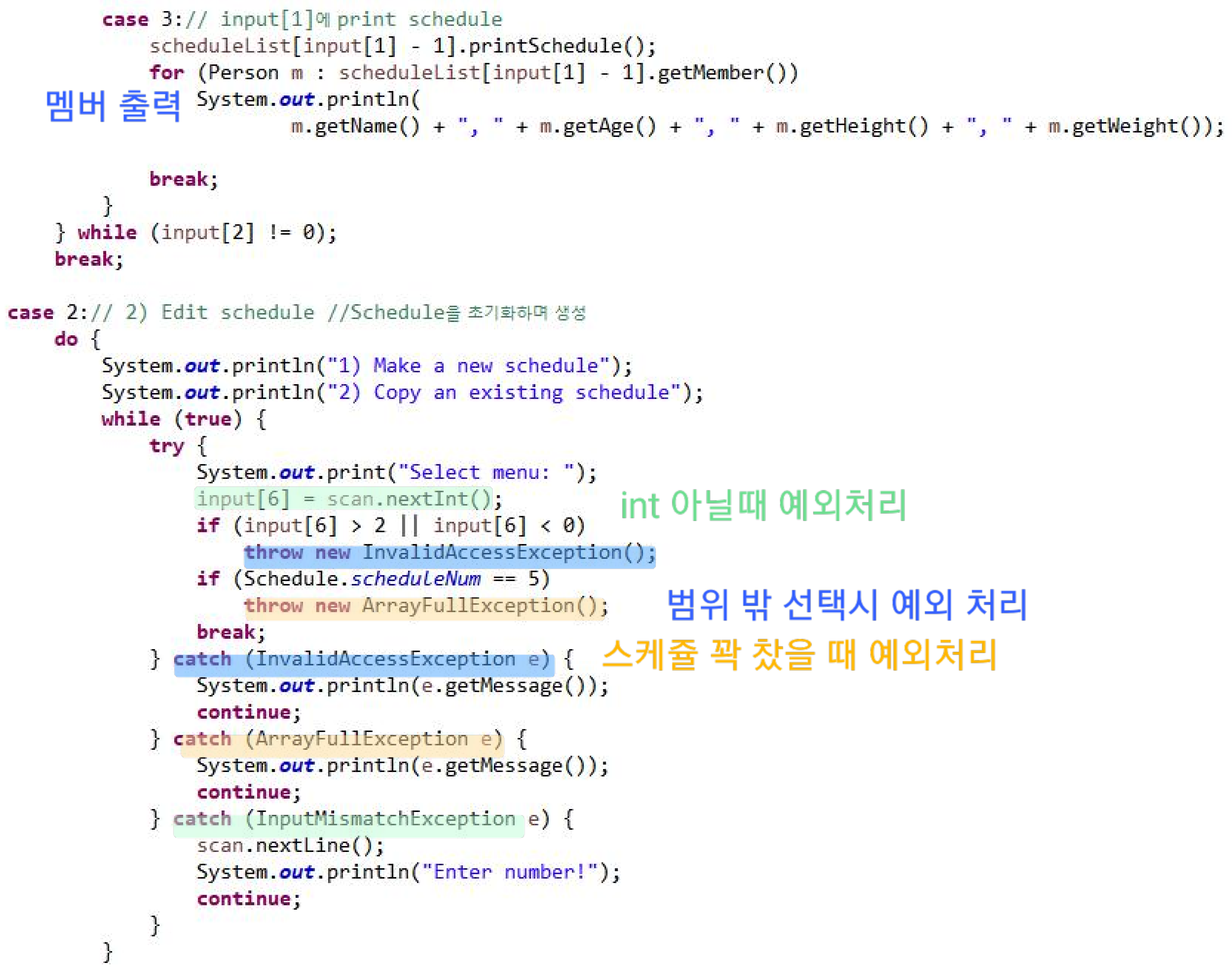
멤버의 신장,체중 中 activity의 최소
신장,체중 이하가 있는 경우 예외처리

s의 type은 ExtremeActivity이므로 Extreme의
getActualPrice()와 binding



```java
case 2:// input[1]에 remove activity
    scheduleList[input[1] - 1].printSchedule();
    int day = 0, time = 0;
    while (true) {
        try {
            System.out.print("Enter the day to remove activity: ");
            day = scan.nextInt();
            if (day > scheduleList[input[1] - 1].getDays() || day < 1)
                throw new InvalidAccessException(" Day");
            break;
        } catch (InvalidAccessException e) {
            System.out.println(e.getMessage());
            continue;
        } catch (InputMismatchException e) {
            scan.nextLine();
            System.out.println("Enter number!");
            continue;
        }
    }
    while (true) {
        try {
            System.out.print("Enter the time to remove activity: ");
            time = scan.nextInt();
            if (time > 20 || time < 9)
                throw new InvalidAccessException(" Time");
            break;
        } catch (InvalidAccessException e) {
            System.out.println(e.getMessage());
            continue;
        } catch (InputMismatchException e) {
            scan.nextLine();
            System.out.println("Enter number!");
            continue;
        }
    }
    try {
        check = scheduleList[input[1] - 1].removePlan(day, time);
        if (check == 1)
            System.out.println("Removed successfully");
    } catch (InvalidAccessException e) {
        System.out.println(e.getMessage());
        continue;
    }
    break;
```

removePlan()의 throws 이용



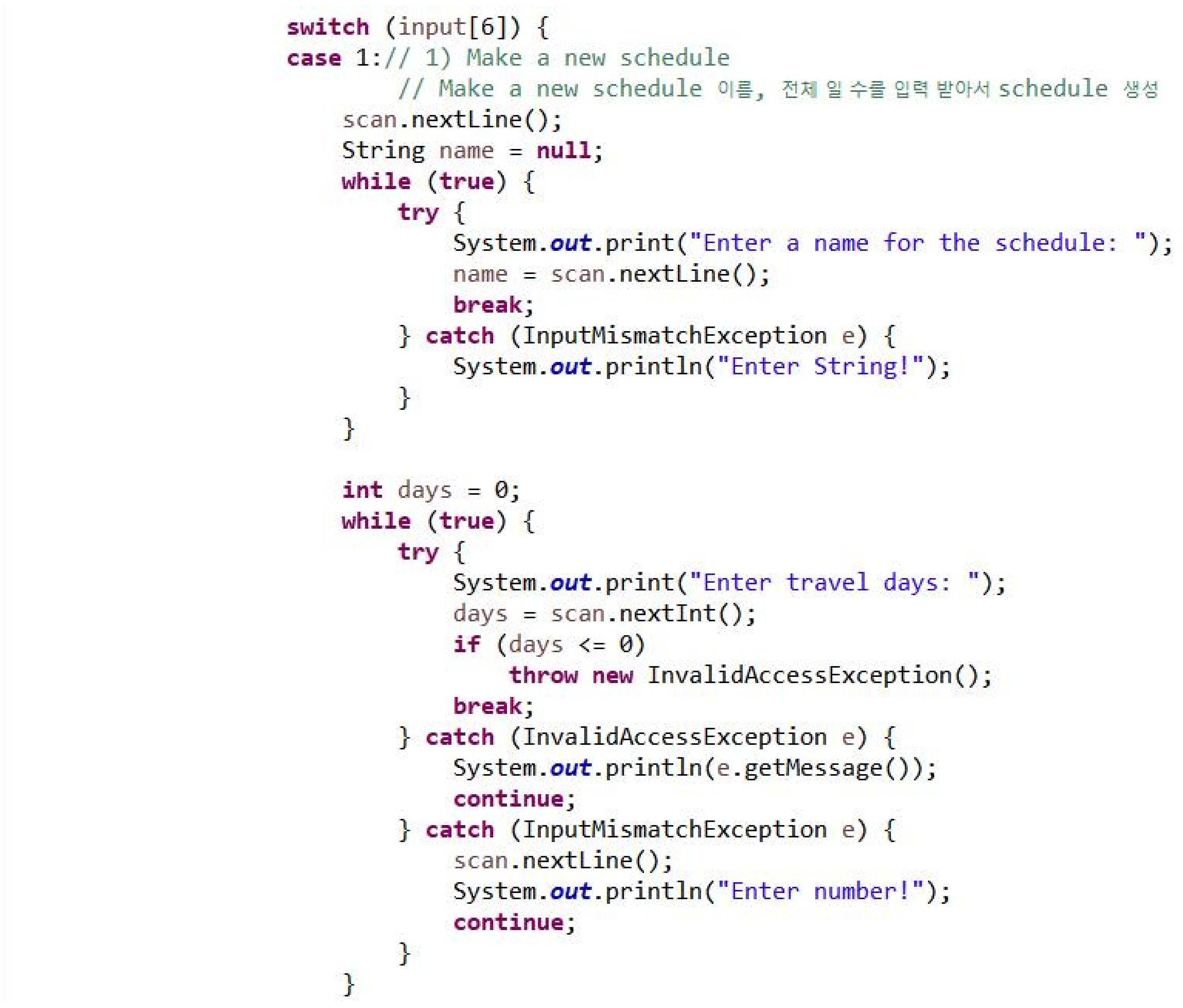
```java
        case 3:// input[1]에 print schedule
            scheduleList[input[1] - 1].printSchedule();
            for (Person m : scheduleList[input[1] - 1].getMember())
                System.out.println(
                        m.getName() + ", " + m.getAge() + ", " + m.getHeight() + ", " + m.getWeight());

            break;
        }
    } while (input[2] != 0);
    break;

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");
        while (true) {
            try {
                System.out.print("Select menu: ");
                input[6] = scan.nextInt();
                if (input[6] > 2 || input[6] < 0)
                    throw new InvalidAccessException();
                if (Schedule.scheduleNum == 5)
                    throw new ArrayFullException();
                break;
            } catch (InvalidAccessException e) {
                System.out.println(e.getMessage());
                continue;
            } catch (ArrayFullException e) {
                System.out.println(e.getMessage());
                continue;
            } catch (InputMismatchException e) {
                scan.nextLine();
                System.out.println("Enter number!");
                continue;
            }
        }
```

멤버 출력

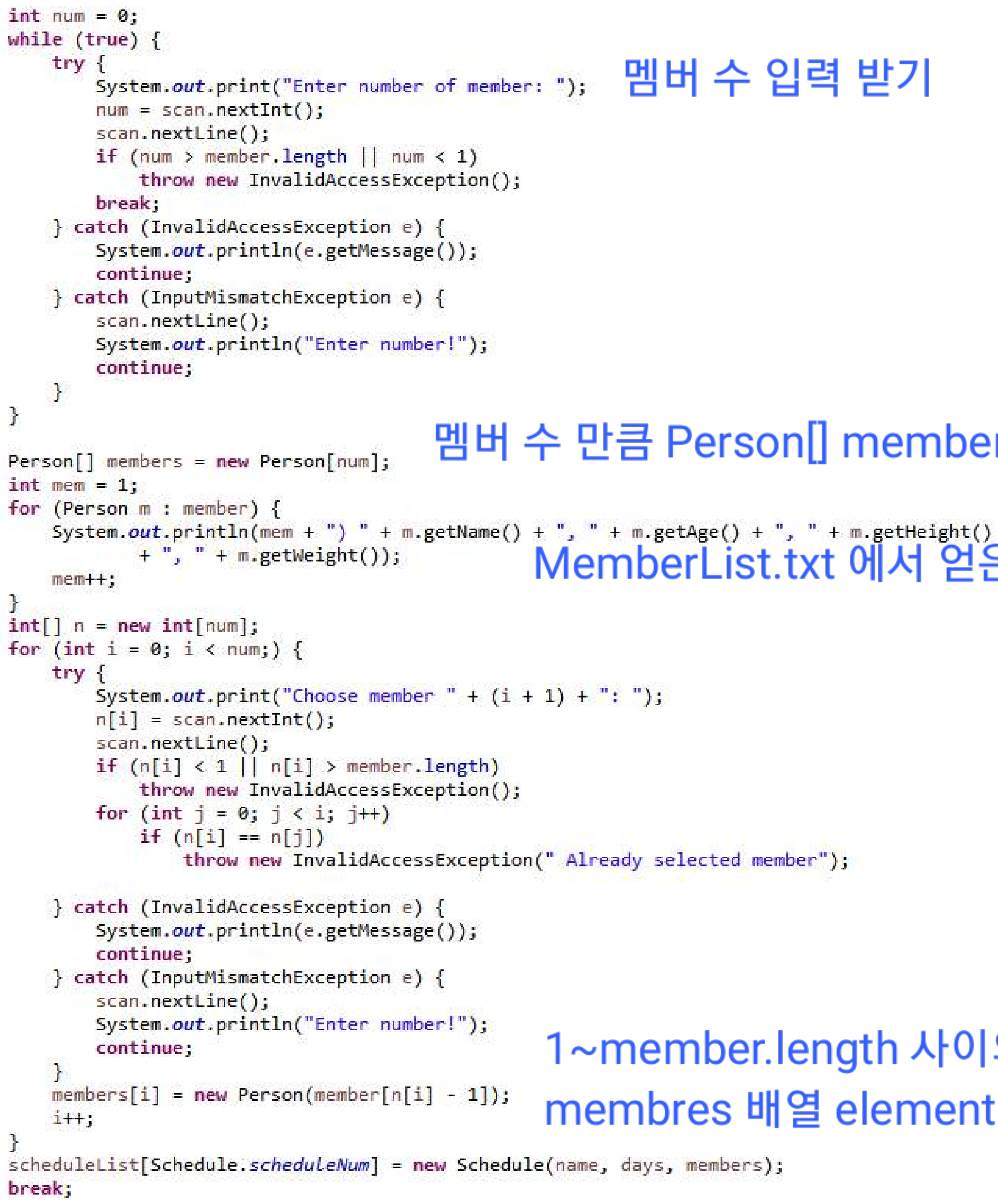
int 아닐때 예외처리

범위 밖 선택시 예외 처리

스케쥴 꽉 찼을 때 예외처리



```java
switch (input[6]) {
case 1:// 1) Make a new schedule
        // Make a new schedule 이름, 전체 일 수를 입력 받아서 schedule 생성
    scan.nextLine();
    String name = null;
    while (true) {
        try {
            System.out.print("Enter a name for the schedule: ");
            name = scan.nextLine();
            break;
        } catch (InputMismatchException e) {
            System.out.println("Enter String!");
        }
    }

    int days = 0;
    while (true) {
        try {
            System.out.print("Enter travel days: ");
            days = scan.nextInt();
            if (days <= 0)
                throw new InvalidAccessException();
            break;
        } catch (InvalidAccessException e) {
            System.out.println(e.getMessage());
            continue;
        } catch (InputMismatchException e) {
            scan.nextLine();
            System.out.println("Enter number!");
            continue;
        }
    }
```



멤버 수 입력 받기

```java
int num = 0;
while (true) {
    try {
        System.out.print("Enter number of member: ");
        num = scan.nextInt();
        scan.nextLine();
        if (num > member.length || num < 1)
            throw new InvalidAccessException();
        break;
    } catch (InvalidAccessException e) {
        System.out.println(e.getMessage());
        continue;
    } catch (InputMismatchException e) {
        scan.nextLine();
        System.out.println("Enter number!");
        continue;
    }
}
```

멤버 수 만큼 Person[] members 생성

```java
Person[] members = new Person[num];
int mem = 1;
for (Person m : member) {
    System.out.println(mem + ") " + m.getName() + ", " + m.getAge() + ", " + m.getHeight()
            + ", " + m.getWeight());
    mem++;
}
```

MemberList.txt 에서 얻은 Person[] member 출력

```java
int[] n = new int[num];
for (int i = 0; i < num;) {
    try {
        System.out.print("Choose member " + (i + 1) + ": ");
        n[i] = scan.nextInt();
        scan.nextLine();
        if (n[i] < 1 || n[i] > member.length)
            throw new InvalidAccessException();
        for (int j = 0; j < i; j++)
            if (n[i] == n[j])
                throw new InvalidAccessException(" Already selected member");

    } catch (InvalidAccessException e) {
        System.out.println(e.getMessage());
        continue;
    } catch (InputMismatchException e) {
        scan.nextLine();
        System.out.println("Enter number!");
        continue;
    }
    members[i] = new Person(member[n[i] - 1]);
    i++;
}
scheduleList[Schedule.scheduleNum] = new Schedule(name, days, members);
break;
```

1~member.length 사이의 올바른 값을 입력 받아
membres 배열 element 객체 생성

```java
                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");
                    }
                    while (true) {
                        try {
                            System.out.print("Select the schedule to copy: ");
                            input[7] = scan.nextInt();
                            scan.nextLine();
                            if (input[7] > scheduleList.length || input[7] < 0)
                                throw new InvalidAccessException();
                            break;
                        } catch (InvalidAccessException e) {
                            System.out.println(e.getMessage());
                            continue;
                        } catch (InputMismatchException e) {
                            scan.nextLine();
                            System.out.println("Enter number!");
                            continue;
                        }
                    }

                    if (input[7] > Schedule.scheduleNum || input[7] == 0)
                        break;

                    String s_1 = null;
                    while (true) {
                        try {
                            System.out.print("Enter a new schedule name: ");
                            s_1 = scan.nextLine();
                            break;
                        } catch (InputMismatchException e) {
                            System.out.println("Enter String!");
                            continue;
                        }
                    }
                    scheduleList[Schedule.scheduleNum] = new Schedule(s_1, scheduleList[input[7] - 1], member);
                    break;
                }
            } while (input[6] != 0);
            // break;

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

    }
    scan.close();
    inputStream.close();
  }
}
```