P3 - Second theoretical work

The method of interest to be tested is “ActivityRecommendation.

Identify the variables for testing

The variables that modifies the behavior of the “ActivityRecommendation” method are:

1. **Weather:**
   - Temperature: Determines the type of activity or if no activity is recommended.
   - Humidity: Used to evaluate conditions for skiing or outdoor tourism.
   - Precipitation: Affects whether outdoor activities are recommended.
   - isCloudy: Impacts recommendations for outdoor tourism.
2. **HealthStatus:**
   - isHealthy: If false, no activity is recommended.
   - hasValidVaccinationCard: If false, no activity is recommended.
3. **SpaceConstraint:**
   - currentCapacity: The number of people currently in the space.
   - maxCapacity: The maximum alowed capacity
   - isWithinCapacity: Determines if certain activities are possible

Variable	Type	Class	Purpose
Temperature	Float	Weather	Determines activity based on temperature ranges.
Humidity	Float	Weather	Impacts skiing and tourism recommendations.
isPrecipitation	Boolean	Weather	Indicates if it is raining; impacts outdoor activities.
isCloudy	Boolean	Weather	Affects outdoor tourism recommendations.
isHealth	Boolean	HealthStatus	Ensures user is healthy for activity recommendations.
hasVaccinationCard	Boolean	HealthStatus	Ensures user has the necessary vaccinations.
currentCapacity	Int	SpaceConstraint	Tracks the current number of people in the space.
maxCapacity	Int	SpaceConstraint	Tracks the maximum allowed capacity for the space.
isWithinCapacity	Boolean	Derived	Determines if the space is undercapacity (currentCapacity < maxCapacity).

Identify test values for variables.

1. **Weather:**

- **Temperature:**

  Equivalence partitioning:

  - Below freezing (<0ºC), "0 is not included": **"-5".**
  - Cool (0-14ºC), "15" is not included: **"10".**
  - Moderate (15-25ºC), from 15 to 25 both included: **"20".**
  - Warm (26-35ºC), from 26 to 35 both included: **"30"**.
  - Hot (>35ºC), "35" is not included: **"40"**.

  Boundary Value Analysis:

  - Lower boundary for cold: **"0".**
  - Transition to moderate: **"15".**
  - Transition to warm: **"25".**
  - Upper limit for drinks recommendation: **"35".**

  Error guessing:

  - Extremely low value: **"-100".**
  - Extremely high value: **"100".**

- **Humidity:**

  Equivalence partitioning:

  - Low (<15%): **"10".**
  - Medium (15-60%): **"50".**
  - High (>60%): **"70".**

  Boundary value analysis:

  - Minimum boundary: **"0"**
  - Boundary for skiing: **"15"**
  - Boundary for tourism: **"60".**
  - Maximum boundary: **"100".**

  Error guessing:

  - Invalid negative value: **"-1".**
  - Out-of-range value: **"101".**

- **Precipitation:**

  Equivalent partitioning:

  - **"True".**
  - **"False".**

  Boundary value analysis:

  - Not applicable for boolean values.

Error guessing:

  - Not applicable for boolean values.

- **isCloudy:**

  Equivalent partitioning:

  - **"True".**
  - **"False".**

  Boundary value analysis:

  - Not applicable for boolean values.

  Error guessing:

  - Not applicable for boolean values.

2. **HealthStatus:**

- **isHealthy**:

  Equivalent partitioning:

  - **"True".**
  - **"False".**

  Boundary value analysis:

  - Not applicable for boolean values.

  Error guessing:

  - Not applicable for boolean values.

- **hasVaccinationCard:**

  Equivalent partitioning:

  - **"True".**
  - **"False".**

  Boundary value analysis:

  - Not applicable for boolean values.

  Error guessing:

  - Not applicable for boolean values.

3. **SpaceConstraint:**

- **currentCapacity:**

  Equivalent partitioning:

  - Under capacity: **"5"** (current < max)
  - At capacitty: **"10"** (current = max)
  - Over capacity: "**15"** (current>max)

  Boundary value analysis:

- Minimum boundarty: **“0”**
- Valid capacity: **“maxCapacity -1”**
- Exact boundary: **“maxCapacity”.**
- Invalid boundary: **“maxCapacity + 1”.**

Error guessing:

- Negative capacity: **“-1”**.

- **maxCapacity:**

  Equivalent partitioning:

  - Small capacity: **“5”**
  - Moderate capacity: **“10”** (current = max)
  - Large capacity: “**20”** (current>max)

  Boundary value analysis:

  - Invalid minimum boundarty: **“0”**
  - Valid case: **“currentCapacity+1”**
  - Edge case: **“currentCapacity”.**

  Error guessing:

  - Negative capacity: **“-1”**.

Variable	EP Values	BVA Values	Error guessing
Temperature	-5, 10, 20, 30, 40	0, 15, 25, 35	-100, 100
Humidity	10, 50, 70	0, 15, 60, 100	-1, 101
IsPrecipitation	True, false	-	-
isCloudy	True, false	-	-
isHealthy	True, false	-	-
HasValidVaccinationCard	True, false	-	-
currentCapacity	5, 10, 15	0, max-1, max, max+1	-1
maxCapacity	5, 10, 20	0, current, current+1	-1

Calculate the maximum possible test cases.

For each variable:

- Temperature**: 11 values**.
- Humidity: **9 values**.
- isPrecipitation: **2 values**.
- isCloudy: **2 values**.
- hasVaccinationCard: **2 values**.
- isHealthy: **2 values**.
- currentCapacity: **8 values.**
- maxCapacity: **7 values**.

Including currentCapacity and maxCapacity explicitly adds detail for edge cases in the capacity-related logic (e.g., near and equal thresholds). This approach is helpful for thorough testing of boundary and error conditions but increases complexity significantly.

Using combinatorial testing:

Maximum Test Cases = 11x9x2x2x2x2x8x7= 88704 test cases.

Define a minimal set of test cases.

Using the **"**each-use" approach (ensuring each value is tested at least once):

Test Case	Temp.	Humid.	Precip.	Is Cloudy	Is Healthy	hasVaccination Card	Current Capacity	maxCapacity
1	-5	10	True	False	True	True	5	Current+1
2	10	50	False	False	True	True	10	current
3	20	70	False	False	True	True	15	0
4	30	0	False	True	True	True	0	5
5	40	15	False	False	True	True	Max-1	10
6	0	60	True	True	False	True	Max	15
7	15	100	False	False	True	False	Max+1	15
8	25	-1	True	False	True	True	-1	10
9	35	101	False	True	False	False	10	-1
10	-100	50	True	False	True	True	15	15
11	100	70	False	False	False	False	0	5

Define sets of tests to achieve pairwise coverage using the algorithm explained in class.

Temperature: -5, 10, 20, 30, 40, 0, 15, 25, 35, -100, 100

Humidity: 10, 50, 70, 0, 15, 60, 100, -1, 101

IsPrecipitation: True, False

IsCloudy: True, False

IsHealthy: True, False

hasVaccinationCard: True, False

currentCapacity: 5, 10, 15, 0, max-1, max, max+1, -1

maximumCapacity: 5, 10, 20, 0, current, current+1, -1

Temp.	Humid.	IsPrecipit.	IsCloudy	IsHealthy	hasVaccin.	currentCapacity	maxCapacity
15	70	False	False	False	True	15	20
25	100	True	True	True	False	10	current
30	0	False	True	True	True	5	0
10	70	True	True	False	False	max	10
0	15	True	False	True	False	max-1	current+1
30	100	False	False	False	False	max+1	-1
20	0	True	False	False	False	0	5
30	101	True	True	True	False	-1	20
20	-1	False	True	True	True	max	current+1
40	15	False	True	False	True	max+1	current
20	50	True	True	True	True	10	-1
40	100	True	False	True	False	0	0
10	-1	False	True	False	True	max-1	5
0	101	False	False	False	True	5	10
25	70	False	False	False	True	-1	current+1
-5	60	True	True	True	False	15	5
-100	100	True	False	True	True	-1	10
35	70	False	False	False	True	10	0

40	-1	True	False	True	False	5	-1
-100	50	False	False	False	False	0	current
10	101	True	False	True	True	0	current+1
0	10	True	True	True	False	max+1	20
15	0	True	False	True	False	max	current
20	60	False	False	False	True	5	20
20	10	False	False	False	True	15	10
100	101	True	True	True	False	max	0
15	101	False	True	False	True	max+1	5
0	0	False	True	True	False	-1	-1
10	0	False	True	True	True	10	20
-5	0	False	False	False	True	max+1	10
0	100	True	False	False	True	max	5
25	50	True	False	True	False	max-1	0
-5	50	False	False	True	False	5	current+1
-100	0	True	True	False	False	15	current+1
100	100	False	False	False	True	max-1	20
30	10	True	False	True	True	max-1	current
20	15	False	False	False	False	-1	0
-5	10	True	True	True	True	0	-1
100	60	True	False	True	True	-1	current
-100	101	False	True	True	False	max-1	-1
25	101	False	True	True	False	15	10
15	100	False	True	True	False	10	current+1
25	60	True	False	True	False	max	-1
0	70	False	False	True	True	15	current
35	0	True	True	True	False	max-1	5
-5	15	False	False	False	True	max	20
30	15	False	False	True	False	10	10
40	0	False	False	True	False	-1	5

25	-1	True	False	True	False	0	20
100	50	False	True	True	False	5	5
-5	100	False	False	False	False	15	0
30	-1	True	True	False	True	15	10
35	-1	False	True	True	True	max+1	current
35	101	True	False	True	True	0	10
40	101	True	False	True	True	max-1	10
100	10	False	True	True	False	10	current+1
15	50	False	False	False	True	-1	10
10	10	False	False	True	False	-1	0
100	70	False	True	False	True	15	-1
30	70	True	True	False	True	0	5
30	50	True	False	True	False	max+1	current+1
15	60	True	True	False	True	max+1	0
40	10	False	True	True	False	max	5
25	0	True	True	False	True	10	5
15	15	False	False	True	False	5	-1
-5	101	True	True	True	False	10	current
15	-1	False	False	False	True	-1	0
10	60	True	False	True	True	5	current
40	50	True	False	False	True	15	20
-100	10	True	True	True	True	5	5
-5	70	True	False	False	True	max-1	20
35	10	True	True	True	False	5	-1
25	15	True	False	False	True	max+1	5
0	-1	True	False	True	True	10	0
15	10	False	False	True	False	0	current
-100	70	True	False	False	False	max+1	0
100	15	True	True	True	False	0	10
20	100	True	False	False	False	max+1	current

35	15	False	True	True	True	15	current+1
30	60	False	False	True	False	10	10
100	-1	False	True	True	False	max+1	20
20	101	False	True	False	False	max-1	5
-100	60	True	False	False	False	0	current+1
-5	-1	False	True	True	False	-1	10
0	60	True	False	False	True	0	current+1
35	100	False	True	False	True	max	20
0	50	True	False	True	False	max	-1
-100	15	False	False	True	False	10	20
35	60	True	True	True	True	-1	0
35	50	True	True	False	False	5	5
40	70	True	True	True	True	5	current+1
25	10	False	False	True	True	5	current
10	50	True	False	False	True	max+1	-1
10	100	True	True	False	True	5	20
20	70	True	True	True	True	0	-1
40	60	True	True	True	True	10	current
10	15	True	False	False	True	15	-1
30	-1	True	False	True	False	max	current
-100	-1	True	True	False	True	max	current
100	0	False	True	True	True	15	-1
15	60	True	False	True	True	max-1	current

For code snippets that include decisions, propose a set of test cases to achieve decision coverage.

Test Case	Temp.	Humid.	Precip.	Is Cloudy	Is Healthy	hasVacc.	Current Capacity	Max Capacity	Expected Output
1	10	50	False	False	False	true	5	10	No acitvity recommended
2	10	50	False	False	True	false	5	10	No acitvity recommended
3	-5	10	True	False	True	True	5	10	Stay home
4	-5	10	False	False	True	True	5	10	Go skiing
5	10	50	False	False	True	True	5	10	Go hiking
6	25	70	False	False	True	True	5	10	Outdoor turism
7	25	70	True	False	True	True	5	10	No specific recommendation
8	35	50	False	False	True	True	5	10	Go for drinks
9	40	50	False	False	True	True	5	10	Go to the beach or pool
10	31	50	False	False	True	True	10	10	No specific recommendation

For code snippets that include decisions, propose a set of test cases to achieve MC/DC coverage.

MC/DC Coverage for Decision 1:

if (!healthStatus.isHealthy() || !healthStatus.hasValidVaccinationCard())

Case	isHealthy	*hasValidVaccinationCard*	*ExpectedOutcome*
1	*False*	*True*	*No activity recommended*
2	*True*	*False*	*No activity recommended*
3	*True*	*True*	*No specific recommendation (doesn`t reject activity)*

MC/DC Coverage for Decision 2:

if (weather.getTemperature() < 0)

Case	Temperature	*Expected outcome*
4	*-5*	*Stay Home*
5	*10*	*No specific recommendation.*

MC/DC Coverage for Decision 3:

if (weather.getHumidity() < 15)

Case	Humidity	*Expected outcome*
6	*10*	*Stay Home / Go skiing*
7	*20*	*No specific recommendation*

MC/DC Coverage for Decision 4:

if (weather.isPrecipitation())

Case	isPrecipitation	*Expected outcome*
8	*True*	*Stay Home*
9	*False*	*Go skiing*

MC/DC Coverage for Decision 5:

if (spaceConstraint.isWithinCapacity())

We will test both cases: when the capacity is within limits and when it is not.

Case	currentCapacity	*MaximumCapacity*	*ExpectedOutcome*
10	*5*	*10*	*Go skiing*
11	*10*	*10*	*No specific recommendation*

MC/DC Coverage for Decision 6:

else if (weather.getTemperature() < 15 && !weather.isPrecipitation())

We will test both cases: when the capacity is within limits and when it is not.

Case	temperature	*precipitation*	*ExpectedOutcome*
12	*5*	*False*	*Go hiking*
13	*10*	*true*	*No specific recommendation*
14	*20*	*False*	*No specific recommendation*

MC/DC Coverage for Decision 7:

else if (weather.getTemperature() < 15 && !weather.isPrecipitation())

We will test both cases: when the capacity is within limits and when it is not.

Case	temperature	*precipitation*	*isCloudy*	*Humidity*	*ExpectedOutcome*
15	*25*	*False*	*False*	*70*	*Outdoor tourism.*
16	*25*	*False*	*False*	*50*	*No specific recommendation*
17	*25*	*True*	*False*	*70*	*No specific recommendation*
18	*25*	*False*	*True*	*70*	*No specific recommendation*
19	*10*	*False*	*False*	*70*	*Go hiking*

Case 15 tests when all conditions are true.

Case 16 tests when humidity is not greater than 60.

Case 17 tests when there is precipitation.

Case 18 tests when the weather is cloudy.

Case 19 tests when the temperature is below 25.

Comment on the results of the number of test cases achieved in sections 4, 5 and 6. What could be said about the coverage achieved?

As we got a maximum 88704 test cases in section 4, the coverage in:

- Section 5: 11/88704 -> 0.0124% of coverage
- Section 6: 100/88704 -> 0.113% of coverage