Exercise Session:
Test Case Design

Gregory Gay
DIT636/DAT560 - February 6, 2025

Airport Connection Check

- API function to check validity of a connection between two flights.
  - If the arrival airport of Flight A differs from the departure airport of Flight B, connection is invalid.
  - If departure time of Flight B is too close to the arrival time of Flight A, connection is invalid.
  - If an airport doesn't exist, the connection is invalid…

Airport Connection Check

`validConnection(Flight flightA, Flight flightB)`
`                returns ValidityCode`

A **Flight** is a data structure consisting of:
- A unique identifying flight code
  (string, three characters followed by four numbers).
- The originating airport code (three character string).
- The scheduled departure time from the originating airport
  (in universal time).
- The destination airport code (three character string).
- The scheduled arrival time at the destination airport (in universal time).
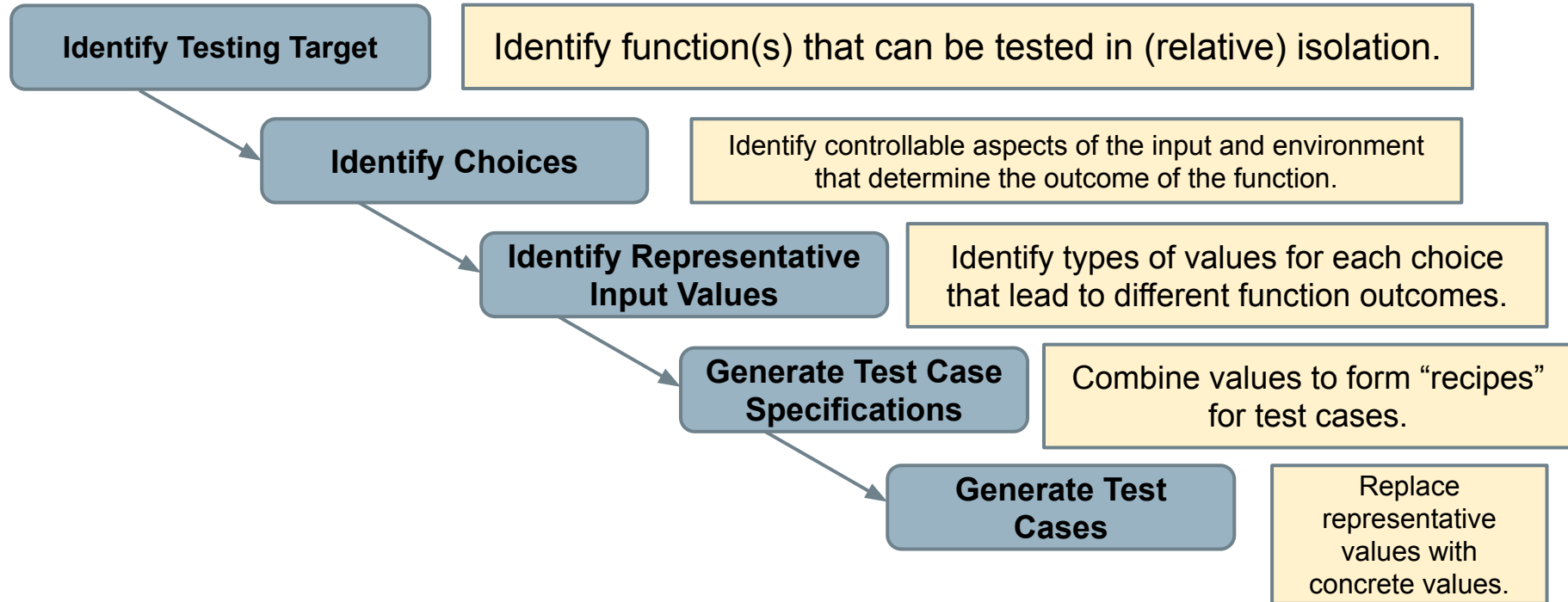
Airport Connection Check

There is also a flight database, where each record contains:
- Three-letter airport code (three character string).
- Airport country (string).
- Schengen Area (boolean).
- Minimum domestic connection time
  - (integer, minimum num. minutes that must be allowed for flight connections to be valid).
- Minimum international connection time
  - (more time is required due to need to clear customs and meet regulations)

ValidityCode is an integer with value:
- 0 for OK
- 1 for invalid airport code
- 2 for a connection that is too short
- 3 for flights that do not connect (flightA does not land in same location as flightB)
- 4 for any other errors (malformed input or any other unexpected errors).

Creating Test Cases

Identify Testing Target	Identify function(s) that can be tested in (relative) isolation.
Identify Choices	Identify controllable aspects of the input and environment that determine the outcome of the function.
Identify Representative Input Values	Identify types of values for each choice that lead to different function outcomes.
Generate Test Case Specifications	Combine values to form "recipes" for test cases.
Generate Test Cases	Replace representative values with concrete values.

Your Task

`validConnection(Flight flightA, Flight flightB)`

`returns ValidityCode`

Identify Choices

Identify controllable aspects of the input and environment that determine the outcome of the function.

Identify Representative Input Values

Identify types of values for each choice that lead to different function outcomes.

Apply Constraints

ERROR, SINGLE, IF

Hints

- Two explicit parameters (Flight A and B) and one implicit (airport database).
  - Flight has multiple fields (potential choices)
  - Database records have multiple fields (potential choices).
  - Remember that representative values can interact. This must be accounted for.
    - **IF constraints indicate when combinations of values should be used for different choices.**

Hints

- Consider how arrival time (flight A), departure time (flight B), and minimum connection time interact.

- Consider that domestic and international connection times can differ in length.

- Consider how the database contents can influence behavior.

- Consider how input can be invalid or malformed
  - (don't just list "invalid input" but give clear examples).

Example to Start

FlightA

Choice: Originating Airport Code

- Valid airport
- Not in database **[error]**
- Not a correctly formatted airport **[error]**
  - (not a three-letter string)

FlightB

Choice: Originating Airport Code

- Valid airport, same as FlightA's Destination Airport Code
- Valid airport, but different from FlightA's Destination Airport Code **[error]**
- Not in database **[error]**
- Not a correctly formatted airport **[error]**