Lab 3: Creating Test Cases Using the Category Partition Method (10 points)

CSE 2102 Software Engineering (Spring 2024)

IMPORTANT DATES

Wednesday (3/29/2024)	Lab3 released
Friday (4/12/2024)	Team progress assessment on GitHub's issue page (*before 5pm*)
Friday (4/19/2024)	Lab3 due (*before 11:59pm*)

TA Hours are during the lab sections. If you want to ask for TA support beyond these hours, feel free to email them (emails are listed on syllabus)

PREPARATION INSTRUCTIONS:

All the students must **work in teams** to carry out Lab3.

Executing the TSL tool:

1. Download the TSL tool (tsl.zip) from HuskyCT -> lab 3 and unzip it
2. Read the TSL tool documentation (tsl-usage.pdf, tsl-tools-spec.pdf).
3. Use the Makefile in the directory to build the TSL tool
4. Make sure you can run the TSL tool on the "findspec" example that is provided. Note: you may need to ensure that the executable is someplace that is on your "path" – google "setting path on linux" if you aren't familiar with this.
5. Make sure you know how to write a TSL specification and apply the TSL tool to it, to generate test frames.

TASK DESCRIPTION:

For the ease of grading, **use your GitHub Repository created for Lab1**, but create a new "Lab3" folder and upload your files under the "Lab3" folder.

Navigate your web browser to the Delta Airlines flight search web page:

https://www.delta.com/flightsearch/book-a-flight

Here you will find a web form with several fields.

Your task is to create a TSL specification for this web form, using appropriate categories and choices and adding constraints as necessary. Your final specification should yield a "reasonable" number of frames, e.g., a number that could feasibly be turned into actual test cases with a day of effort. A range between 100 and 300 test frames would be considered acceptable.

We're going to get you started by suggesting/requesting a few specific things and setting some limits. We'll then ask some specific questions.

As a first limit, limit yourselves to the Round Trip option, since the form doesn't change much for One-Way and has considerably more complexity for the Multi-City option that isn't necessary for the lab. Also, ignore page elements that provide links to information (for example, the link entitled `` Use Certificates, eCredits, or Delta Gift Cards''). Other than these, your final TSL specification must account for every page element that can have an effect on the final form that gets submitted when you click on "Find Flights".

Since the number of airports is large, limit your TSL specification to two valid departure locations and three valid arrival locations (though you should add a choice corresponding to invalid locations to each).

For departing locations use:

1. BDL # Hartford/Springfield, CT
2. PVD # Providence, RI

For arriving locations use:

1. ATL # Atlanta, GA
2. EWR # Newark New Jersey
3. MSP # Minneapolis/St Paul, MN, for testing the non-stop

Departure and arrival date information is a required field on the form so you will need to generalize these inputs by using categories and choices. If you use their date selector you will discover that there is an upper limit enforced on both the departure and return dates that is approximately one year in the future. Limit your choices to the following:

- Today
- 1 month in the future
- 11 months in the future

Since it is possible to enter dates by hand rather than by using the calendar, several error conditions would be interesting to test. What are these error conditions that you can envision? Represent these in your TSL specification.

Beyond the fields just discussed, several others have large numbers of possible selections:

- Departure and arrival airports
- Departure time
- Return time
- Number of passengers
- Search options
- Show fares
- Best fares for

For these, you will need to make choices that yield reasonable "partitions" to reduce the numbers of tests. But you will also need to use constraints to reduce the number of combinations that need to be considered.

With respect to some other fields, here are some questions.

1. The meeting event code field is optional. How can you handle this field in TSL?
2. Notice that the some fields (e.g., "My dates are flexible" and "Include Nearby Airports") are mutually exclusive. How can you use the [property] and [if] features of the TSL specification language to reflect this?
3. When the Shop with Miles is selected, how does the form change? How may you reflect this in the TSL?

GRADING (10 POINTS IN TOTAL)

- Before 5pm, Friday (4/12/2024), **each team** must report their Lab3 progress—in terms of (1) completion percentage (e.g., 20%, 50%, 85%, etc.), and (2) each team member's contributions (or lack thereof)—on the "Issues" page of the team's repository on GitHub. All the teams are strongly encouraged to be transparent on the "Issues" page throughout Lab3; however, providing a team-wide progress update before 5pm, 4/12/2024 is mandatory. (**2 points**)

- Final deliverable in the "lab 3" folder
    - Your completed TSL specification (**2 points**)
    - Your generated test frames (**2 point**)
    - A report (in the format of plain text) indicating:
        - The number of "frames" that the TSL tool reported on that specification (**1 point**)
        - Answers to the three questions above (your answers should describe things that you included in your TSL spec to handle the four issues.) (**3 points**)

- Before 11:59pm, Friday (4/19/2024), **each student** must upload a completed "Lab3-SubmissionFile" to HuskyCT as the submission to Lab3.

NOTES

- Missing or late submissions will receive 0 automatically.
- Each student's Lab3 grade will be calculated as follows:

Individual grade = Team grade * Individual contribution coefficient