Homework: Test Techniques I

1. Equivalence Partitioning / Boundary Value Analysis – Income Checker

Now that you are familiar with the Equivalence Partitioning / Boundary Value Analysis Techniques, let's recall **The Income Checker App** from the QA Basics course, that categorizes the given **monthly income** into one of the following categories: "**low**", "**mid**", "**high**". It works as follows:

- If the income is less than 1000, returns "**low**"
- If the income between 1000 (inclusively) and 3000 (exclusively), returns "**mid**"
- If the income is equal or bigger than 3000, returns "**high**"
- If the income is negative, returns "**error**"

Your task is:

Equivalence Partitioning: Divide the possible input values of the "**income**" into different equivalence classes or partitions. Remember to include both valid and invalid partitions.

Boundary Value Analysis: Identify the boundary values of the defined partitions and come up with test cases that include these boundary values. Ensure you consider "**edge cases**" - values just outside of valid ranges.

Note: Keep in mind that testing should cover not only expected or valid inputs but also unexpected or invalid ones. Consider all possible scenarios that might be encountered in a real-world situation.

Equivalence Partitioning Test Cases including invalid cases:

Test Case ID	Input	Expected Output
TC01	-10	"error"
TC02	500	"low"
TC03	2000	"Mid"
TC04	4000	"High"

Boundary Value Analysis Test Cases including invalid cases:

Test Case ID	Input	Expected Output
TC01	-1	"error"
TC02	0	"Low"
TC03	1	"low"
TC04	999	"low"
TC05	1000	"Mid"
TC06	1001	"Mid"
TC07	2999	"Mid"
TC08	3000	"High"
TC09	3001	"Hihg"

2. Pairwise Testing - eCommerce Checkout Function

Assume you have a checkout function of an eCommerce application for testing. The function contains the following fields with their input values:

Drop-down menu that contains 5 different shipping methods (input values – 1, 2, 3, 4, 5);

Radio button for gift wrapping (input values – Yes or No);

Checkbox for agreeing to terms and conditions (input values - Checked or Unchecked);

Place Order button (input values - Does not accept any value, only finalizes the order).

Your task is:

1. Calculate how many would be the positive test cases, if you have to cover every single possibility?

Your Answer: 20

Using Pairwise testing, reduce the number of necessary test cases.

Add a screenshot of the reduced test cases here

	Shipping methods	Gift wrapping	Terms and conditions
1	1	Yes	Checked
2	1	No	Unchecked
3	2	No	Checked
4	2	Yes	Unchecked
5	3	No	Checked
6	3	Yes	Unchecked
7	4	Yes	Checked
8	4	Yes	Unchecked
9	4	No	Checked
10	5	Yes	Unchecked
11	5	Yes	Checked
12	5	No	Checked

We have only considered positive test cases so far. What about negative ones? Write at least 3 negative test cases.

Attempt to place an order with no shipping method selected.
Attempt to place an order without agreeing with the Terms and Conditions
Choose an invalid value for the gift wrapping option, such as selecting both "Yes" and "No" simultaneously or choosing a value that is not "Yes" or "No