Exercise 05: Scoreboard implementation

Objective: *Introduction to the UVM Scoreboard component.*

Task: *Implementation of the Scoreboard component to compare the output of the Saturation Filter with the Reference Model results.*

The results from the DUT and the Reference Model need to be compared to ensure the correct behavior of the design. In order to compare, these results must be stored and share between the testbench components. For that purpose, queues are used.

The DUT's input is sent via the Producer Agent's analysis port to a FIFO in the Reference Model. Then, the output of the Reference Model is sent through the analysis port to a FIFO in the Scoreboard. Through the Consumer Agent, the DUT's output is then shared via the analysis port to another FIFO in the Scoreboard. In the Scoreboard the content of the FIFOs is copied to separated queues which are then used to compare the results.

A simple implementation of the Reference Model code can be found in `<ROOT>/sat_filter/src/tb/ref_model` and the UVM class can be found in `<ROOT>/sat_filter/src/tb/sat_filter_ref_model.py`.

A skeleton of the Scoreboard class is implemented in the file, `<ROOT>/sat_filter/src/tb/sat_filter_scb.py`. Notice that all the required components are already implemented in the `build_phase`.

The Scoreboard implementation must be completed as follows:

1. Implementing the missing steps to get the items from the FIFOs (uvc_ssdt_consumerfifo, ref_model_fifo) and into the Queues (uvcs_ssdt_consumer_queue, ref_model_queue). **HINT:** Make a coroutine which calls get on the FIFO and put on the queue.
2. Then compare against each other. If the items are matching, the `success` variable must be increased, if not an error message must be printed and the `failure` variable increased. **HINT:** Extend the run_phase with a never ending loop which calls get on the two queues and compares.
3. Implement the `check_phase` to print the value of the `success` and `failure` variables.

Now the Scoreboard must be integrated in the testbench, specifically in the `environment` component, located in `<ROOT>/sat_filter/src/tb/sat_filter_tb_env.py`. For that, the next steps must be followed:

1. First, the Scoreboad handler must be included in the class constructor, then instantiated in the `build_phase`.
2. Connect the Consumer Agent's analysis port with the Scoreboard consumer's FIFO, created before.
3. Connect the Reference Model's analysis port with the Scoreboard reference model's FIFO, created before.

After the scoreboard is integrated into the testbench, check if both the `success` and `failure` vairables correctly count up for matching and mismatching items respectively. Consider also

how to test if the scoreboard catches mismatched items. For example, inject transactions with errors into the scoreboard to see if it catches them. This is also known as robustness testing. A possible way to achive this would be, to modify the DUT to always send 0 as output data.