Outline of the process by which alternatives are generated and determined feasible or infeasible within the Python MASR program

General flow of processes in the alternatives.py module (i.e., how the Python program generates a list of feasible alternatives):

The process of generating alternatives starts when the user presses the "Find Alternatives" button on the main GUI. At this point, all mission, manufacturing, and performance requirements are collected by the program and saved in a list. In addition, the user-specified importance weightings of the performance metrics are saved to be used in the TOPSIS algorithm later in the process. Next, the *alternatives.generate_alternatives* function is called, which uses the parts stored in the databases to compute a full-factorial list of all possible alternatives, both feasible and infeasible. After each alternative is constructed from some combination of COTS parts, that alternative's attributes are passed to the *alternatives.is_feasible* function. The *alternatives.is_feasibl*e function uses the alternative's attributes to compute the alternative's size and performance. If the size and performance of the alternative satisfies the user-specified constraints, then the alternative is deemed "feasible". If the size or performance characteristics of the vehicle alternative do not satisfy the constraints, then the alternative is deemed "infeasible". After the list of all alternatives are either deemed feasible or infeasible, the list is sent back to the function that originally called *alternatives.generate_alternatives*. Finally, a list of feasible alternatives is constructed from the full list of alternatives and each feasible alternative is scored based on the user-specified performance metric importance weightings. The result is a list of feasible vehicle alternatives that can be sorted not only by a specific performance metric (e.g., endurance, payload, etc.), but also by the TOPSIS score.

More detailed description of sizing and performance calculations in the *alternatives.is_feasible* function (see code comments for more specific details):

The *alternatives.is_feasible* function takes in the attributes of a vehicle as well as user-specified mission, manufacturing, and performance constraints and determines whether the alternative is able to fulfill the mission. A vehicle alternative that can perform the required mission is called "feasible". A vehicle alternative that does not satisfy one or more constraints is called "infeasible".

- First, the vehicle size is estimated based on battery size, propeller size, and motor size. After the size is estimated the following size constraints are applied
    - The maximum vehicle dimension must be less than the maximum size constraint.
    - The maximum dimension of the hub must be less than the maximum dimension of the laser cutter. In the current model the hub layers are cut using a laser cutter.
    - The length of the arm must be less than the maximum dimension of the 3D printer. This assumes the other dimension of the printer is sufficiently large, which is a fair assumption since the arm is long and narrow. This also assumes the printer height is sufficient.

- The function *hublayout.hub_layout* is called to size the hub for the chosen battery, sensors, and electronic component set. See comments in the *hublayout.py* module for more information on the hub sizing function.
- Next, the total weight of the vehicle is estimated based on the vehicle geometry and known material densities, as well as the known weights of hardware, sensors, and other electronic components.
    - The vehicle weight constraint is now checked to make sure the vehicle is not too heavy.
- Next, the thrust available for the vehicle is estimated based on the maneuverability requested by the user and the vehicle weight that was just calculated.
    - The payload capacity requirement is now checked by comparing the payload required to the power available during hover at maximum throttle.
- The estimated vehicle endurance is estimated and compared to the endurance required by the user. The equation for average thrust used in the code assumes that the mission consists only of hovering. This could later be replaced with a more accurate mission model.
    - The vehicle endurance constraint is checked
- The estimated build time is calculated using an equation from the original MASR Excel tool.
    - The estimated build time constraint is checked.
- Now that all of the constraints have been checked, if the vehicle alternative has not failed any constraint checks, it is deemed feasible and its performance characteristics are saved.