# TSA Region 8

# Coding Problem 2018

# Rocket Team

## Background

After graduating high school with stellar TSA skills, you have the opportunity to join the MIT Rocket Team. This year, the team hopes to fly an oversized rocket in the deserts of New Mexico. To ensure the avionics package functions before placing it on the rocket, your first task is to design a streamlined, easily testable avionics codebase.

## Requirements

The rocket must move sequentially between the Launch, Drogue, Main, and Landed states and print out when it reaches each (these should be separate classes so other behaviors can be added).

The rocket begins in the Launch state (v>0 ft/s).

To enter the Drogue state and safely deploy the drogue parachute, the rocket must ensure it is descending (v<0 ft/s) or, in the case of barometer failure, a failsafe time has been reached (t>15s).

To enter the Main state and deploy the main parachute, the rocket must be low enough to ensure it doesn't drift too far (h<200ft).

To enter the Landed state and activate the recovery GPS, the rocket must be stationary on the ground (v=0 ft/s).

For safety from sensor errors, these conditions must occur 3 cycles in a row.

The rocket has access to a global function *altitude()* returning the current altitude in flight. By saving the current time/altitude at each call and taking the change in altitude over the change in time, you must implement a *velocity()* function.

For simulation purposes, the main program should step though lines of a provided .csv data file with columns time and altitude. Using the current line number, both *altitude()* and *velocity()* should be implemented from reading out the data.

In addition, the returned value for altitude should include some random systemic error (<10%)

Aside from the firmware, a testing GUI may be added to display the current time, altitude, velocity, and state with a button to increment the timestep.

Scoring Rubric

| **Item** | **Possible** | **Score** |
|---|---|---|
| Enters states | 10 | |
| Prints states | 5 | |
| velocity() | 10 | |
| Transition conditions | 10 | |
| Counting conditions | 5 | |
| Simulation data read, used | 10 | |
| Error implemented | 5 | |
| Basic user interface/printout | 10 | |
| GUI | 5 | |
| | | |
| Compiles | 10 | |
| Style | 10 | |
| Efficiency | 10 | |
| | | **/100** |