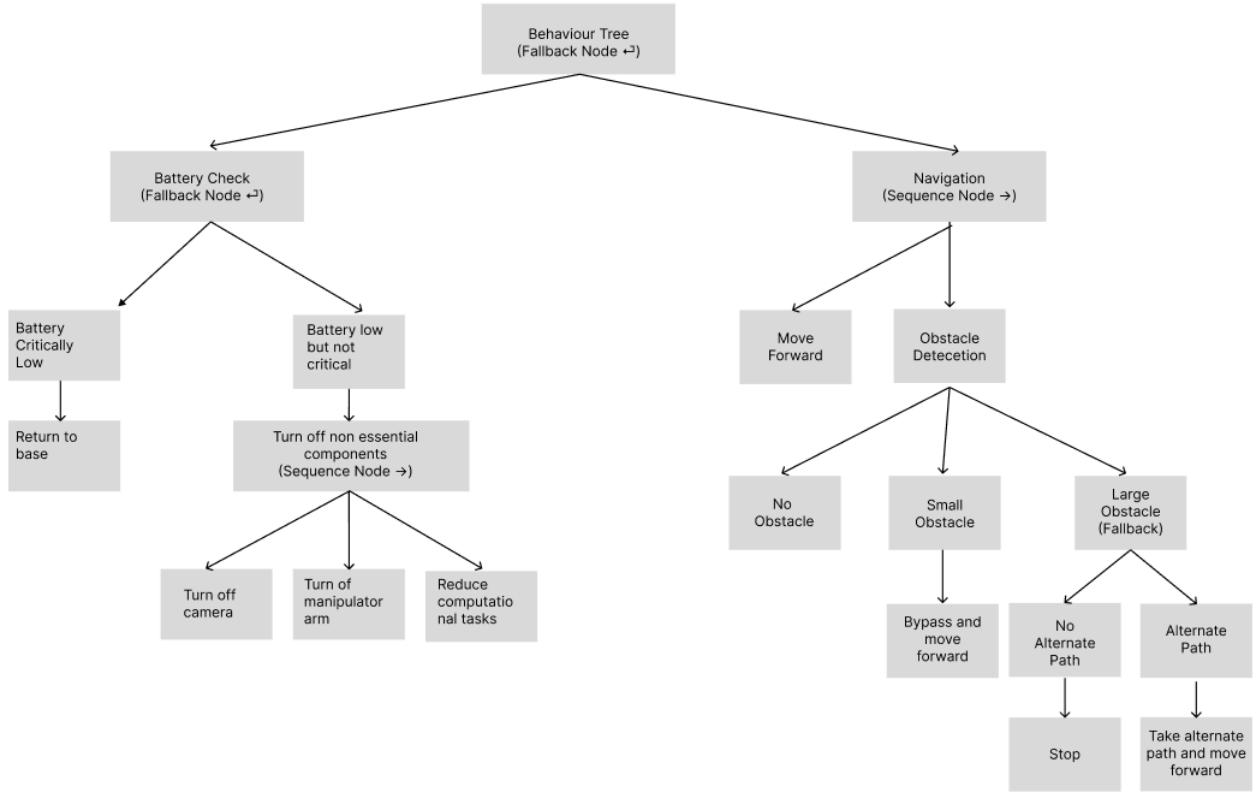
# Mars Task #1
# High Dose Q3

## Behavior Tree Diagram:




Q.) How does the Fallback Node help in making better decisions:
Ans.) In a scenario where multiple tasks have to execute a fallback node goes to all of its children nodes one by one and executes them if all nodes fail it returns failure. Since a fallback node checks all children nodes error management and critical operations such as return to base can be executed smoothly. They also allow to setup priority nodes as it moves from left to right so the most important checks are run first the the less important ones. Like how battery is critically low is run first so that if it is low the rover doesn't waste time and can start proceeding towards the base.

Q.) Why is this better than using long if-else conditions?
Ans .) A Behavior tree provides a visual representation of the tasks making it easier to implement a hierarchical structure which has less faults compared to if-else which has to be written linearly and is is deeply nested making it tough to implement complex task networks. A behavior tree allows us to reuse nodes reducing problem that would come if this were to be done using if-else. IN a BT

a fallback node checks all children node if one fails it goes to the next one programming such using if-else will be tough. Adding new nodes is also very ease in a BT.

Q.) What happens if the battery is low but not critically low?
Ans.) The Battery Check fallback node runs it goes to the first child node Is battery critically low? Which fails it then goes to Is battery low but not critical which is passed then it’s children are executed . If the battery is low but not critically low the rover proceeds to turn off non essential systems such as camera, onboard science equipment, drills and the robotic arm on it so as to save power and reduce power hungry computational tasks in such a way smooth operation of the rover systems still in use are executed properly.

Name : Arpit Srivastava
Roll no : EC24I1001