**ACS script (from ACS.py- DONE):**

(\*\* = repeated command)

(Note: for the longitude check, the user must wait for the random values to change. There is no way to control the longitude directly)

Console: WELCOME TO THE ATTITUDE CONTROL SYSTEMS (ACS) CONSOLE!

Your task is to rotate the satellite for proper payload alignment with the imagery target on the earth’s surface.

\*\*Console: Choose from the following menu of commands:

1. System Checks
2. Verify Alignment
3. CMG Activate Roll
4. CMG Activate Pitch
5. CMG Activate Yaw
6. Transfer Telemetry

**$ 1** {main menu}

Console: Checking Attitude Systems… (5sec pause)

The SimCraft’s current Longitude is [Reached/Not Reached; eta # minutes (true/false)]

The SimCraft’s current Alignment is [Reached/Not Reached (true/false)]

\*\*Console: Choose from the following menu of commands:

1. Longitude Check
2. Verify Alignment
3. CMG Activate Roll
4. CMG Activate Pitch
5. CMG Activate Yaw
6. Transfer Telemetry

**$ 2** {function name: def verifyAlignment)

Console: Verifying Alignment… (5sec pause)

The SimCraft’s Roll Alignment is [Reached/Off by θ°]

The SimCraft’s Pitch Alignment is [Reached/Off by θ°]

The SimCraft’s Yaw Alignment is [Reached/Off by θ°]

\*\*Console: Choose from the following menu of commands:

1. System Checks
2. Verify Alignment
3. CMG Activate Roll
4. CMG Activate Pitch
5. CMG Activate Yaw
6. Transfer Telemetry

**$ 3** {function name: def updateRoll)

Console: How much do you want to change the Roll by (in Degrees)?

**$ *θ*** {input var/type: newRoll/int} (5sec pause)

Console: The SimCraft’s Roll Alignment has changed by θ°

\*\*Console: Choose from the following menu of commands:

1. System Checks
2. Verify Alignment
3. CMG Activate Roll
4. CMG Activate Pitch
5. CMG Activate Yaw
6. Transfer Telemetry

**$ 4** {function name: def updatePitch)

Console: How much do you want to change the Pitch by (in Degrees)?

**$ *θ*** {input var/type: newPitch/int} (5sec pause)

Console: The SimCraft’s Pitch Alignment has changed by θ°

\*\*Console: Choose from the following menu of commands:

1. System Checks
2. Verify Alignment
3. CMG Activate Roll
4. CMG Activate Pitch
5. CMG Activate Yaw
6. Transfer Telemetry

**$ 5** {function name: def updateYaw)

Console: How much do you want to change the Yaw by (in Degrees)?

**$ *θ*** {input var/type: newYaw/int} (5sec pause)

Console: The SimCraft’s Yaw Alignment has changed by θ°

\*\*Console: Choose from the following menu of commands:

1. System Checks
2. Verify Alignment
3. CMG Activate Roll
4. CMG Activate Pitch
5. CMG Activate Yaw
6. Transfer Telemetry

**$ 6** {function name: def telemetryTransfer)

Console: Transferring ACS Telemetry… (5sec pause)

Data has been Transferred! GREAT WORK ON THE ATTITUDE CONTROL SYSTEMS (ACS) CONSOLE!

====================================================

**EPS script (from EPS.py- DONE):**

(\*\* = repeated command)

(note: display solar panel angle in UI)

(note: if you rearticulate the panels after full power, the distribution should be reset to 80% and unable to send telemetry)

Console: WELCOME TO THE ELECTRICAL POWER SYSTEMS (EPS) CONSOLE!

Your task is to command the satellite to full power before the payload can be operated.

\*\*Console: Choose from the following menu of commands:

1. Status Checks
2. Verify Power Distribution System
3. Full Power
4. Articulate Panel
5. Telemetry Transfer

**$ 1** {function name: def systemChecks)

Console: Checking Power Systems… (5sec pause)

The SimCraft’s current Uplink Status is [Reached/Not Reached (true/false)]

The SimCraft’s current Bus Connection Status is [Reached/Not Reached (true/false)]

The SimCraft’s current Articulation System Status is [Reached/Not Reached (true/false)]

If an error occurs, run the command again to refresh and update the system.

{Note (Summer): Have random start booleans and then have some way to make sure they all end up true}

\*\*Console: Choose from the following menu of commands:

1. Status Checks
2. Verify Power Distribution
3. Full Power
4. Articulate Panel
5. Telemetry Transfer

**$ 2** {function name: def verifyPowerDist)

Console: Verifying Power Distribution… (5sec pause)

ACS Distribution… 16%

EPS Distribution… 16%

TCS Distribution… 16%

COMMs Distribution… 16%

Payload Distribution… 16%

Total Distribution… 80%

SimCraft is currently operating at 80% power 100% power is required for this mission activity.

\*\*Console: Choose from the following menu of commands:

1. Status Checks
2. Verify Power Distribution
3. Full Power
4. Articulate Panel
5. Telemetry Transfer

**$ 3** {function name: def fullPower)

Console: Redistributing Resources… (5sec pause)

The current Solar Panel Angle is θ° away from the sun.

Solar panels must be articulated within 10 degrees of the sun angle to increase the total power consumption of the SimCraft.

{Note (Summer): Find variable for sun angle, initial value randomized, articulate command changes the angle}

\*\*Console: Choose from the following menu of commands:

1. Status Checks
2. Verify Power Distribution
3. Full Power
4. Articulate Panel
5. Telemetry Transfer

**$ 4** {function name: def articulate)

Console: How much do you want to articulate the solar panels by (in Degrees)?

**$ *θ*** {input var/type: newAngle/int} (5sec pause)

Console: The SimCraft’s Solar Panel Alignment has changed by θ°

\*\*Console: Choose from the following menu of commands:

1. Status Checks
2. Verify Power Distribution
3. Full Power
4. Articulate Panel
5. Telemetry Transfer

**$ 3** {function name: def fullPower)

Console: Redistributing Resources… (5sec pause)

The current Solar Panel Angle is θ° away from the sun.

Distribution System has routed more power to the ACS and COMMs Subsystems.

SimCraft is now being utilized at 100% power.

\*\*Console: Choose from the following menu of commands:

1. Status Checks
2. Verify Power Distribution
3. Full Power
4. Articulate Panel
5. Telemetry Transfer

======(OPTIONAL; if re-enter 2)=======

**$ 2** {function name: def verifyPowerDist)

Console: Verifying Power Distribution… (5sec pause)

ACS Distribution… 26%

EPS Distribution… 16%

TCS Distribution… 16%

COMMs Distribution… 26%

Payload Distribution… 16%

Total Distribution… 100%

SimCraft is currently operating at 100% power.

=================================

**$ 5** {function name: def telemetryTransfer)

Console: Transferring EPS Telemetry… (5sec pause)

Data has been Transferred! GREAT WORK ON THE ELECTRICAL POWER SYSTEMS (EPS) CONSOLE!

====================================================

**TCS script (from TCS.py- DONE):**

(\*\* = repeated command)

(Note: all temps are taken as the abs val of user input and will always cool the system from a higher temp)

Console: WELCOME TO THE THERMAL CONTROL SYSTEMS (TCS) CONSOLE!

Your task is to perform the cooling procedure for equipment now undergoing thermal exposure in the new attitude position.

\*\*Console: Choose from the following menu of commands:

1. Status Check
2. Verify Thermal Ranges
3. Cool Subsystems
4. Transfer Telemetry
5. Refresh

**$ 1** {main menu)

Console: Checking Thermal Systems… (5sec pause)

The SimCraft’s current Heating Element Status is [Reached/Not Reached (true/false)]

The SimCraft’s current Bus Connection Status is [Reached/Not Reached (true/false)]

The SimCraft’s current Telemetry Signal Status is [Reached/Not Reached (true/false)]

If an error occurs, run the refresh command again to refresh and update the system.

{Note (Summer): Make sure they can all end up true}

**$ 2** {main menu}

Console: Which Subsystem do you want to verify?

1. ACS
2. EPS
3. Payload
4. Comms

**$ 1** {sub-menu}

Console: Verifying ACS Thermal Distribution… (5sec pause)

CMG Thermal (Good) Range… (-)randInt - (+)randInt

Alignment Thermal (Good) Range… (-)randInt - (+)randInt

*{note: user can see real-time temps in UI}*

{Note (Summer): Make up numbers}

**$ 2** {main menu}

Console: Which Subsystem do you want to verify?

1. ACS
2. EPS
3. Payload
4. Comms

**$ 2** {sub-menu}

Console: Verifying EPS Thermal Distribution… (5sec pause)

Power Distribution System Thermal (Good) Range… (-)randInt - (+)randInt

Battery Thermal (Good) Range… (-)randInt - (+)randInt

Articulation System Thermal (Good) Range… (-)randInt - (+)randInt

*{note: user can see real-time temps in UI}*

{Note (Summer): Make up numbers}

**$ 2** {main menu}

Console: Which Subsystem do you want to verify?

1. ACS
2. EPS
3. Payload
4. Comms

**$ 3** {sub-menu}

Console: Verifying EPS Thermal Distribution… (5sec pause)

Optical Electronics Thermal (Good) Range… (-)randInt - (+)randInt

Gimbal System Thermal (Good) Range… (-)randInt - (+)randInt

Imager Thermal (Good) Range… (-)randInt - (+)randInt

*{note: user can see real-time temps in UI}*

{Note (Summer): Make up numbers}

**$ 2** {main menu}

Console: Which Subsystem do you want to verify?

1. ACS
2. EPS
3. Payload
4. Comms

**$ 4** {sub-menu}

Console: Verifying Communications Thermal Distribution… (5sec pause)

On-Board Computer Thermal (Good) Range… (-)randInt - (+)randInt

Signal Processor Thermal (Good) Range… (-)randInt - (+)randInt

*{note: user can see real-time temps in UI}*

{Note (Summer): Make up numbers}

**$ 3** {main menu}

Console: Which Subsystem do you want to cool?

1. ACS
2. EPS
3. Payload
4. Comms

**$ 1**

Console: Which ACS Attribute do you want to cool?

1. CMGs
2. Alignment

**$ 1**

Console: How much do you want to cool the CMGs?

**$ #**

Console: Transferring TCS Telemetry… (5sec pause)

Data has been Transferred! GREAT WORK ON THE THERMAL CONTROL SYSTEMS (TCS) CONSOLE!

====================================================

**Payload script (from Payload.py- DONE):**

(\*\* = repeated command)

(Note: Subsystem has no user input)

Console: WELCOME TO THE PAYLOAD (PL) CONSOLE!

Your task is to capture imagery of the target during the flyover period.

\*\*Console: Choose from the following menu of commands:

1. Status Checks
2. Slew Image
3. Acquire Target
4. Capture Image
5. Transfer Telemetry

**$ 1** {function name: def systemChecks)

Console: Checking Power Systems… (5sec pause)

The SimCraft’s current Optical Electronics Status is [Reached/Not Reached (true/false)]

The SimCraft’s current Bus Connection Status is [Reached/Not Reached (true/false)]

The SimCraft’s current Gimble Connection Status is [Reached/Not Reached (true/false)]

If an error occurs, run the command again to refresh and update the system.

{Note (Summer): Make sure they can all end up true}

\*\*Console: Choose from the following menu of commands:

1. Status Checks
2. Slew Image
3. Acquire Target
4. Capture Image
5. Transfer Telemetry

**$ 2** {function name: def slewImg)

Console: Slew Commencing… (5sec pause)

The imager has reached the ground target area.

\*\*Console: Choose from the following menu of commands:

1. Status Checks
2. Slew Image
3. Acquire Target
4. Capture Image
5. Transfer Telemetry

**$ 3** {function name: def aquireImg)

Console: Acquiring Target… (5sec pause)

The ground target has been acquired successfully.

\*\*Console: Choose from the following menu of commands:

1. Status Checks
2. Slew Image
3. Acquire Target
4. Capture Image
5. Transfer Telemetry

**$ 4** {function name: def captureImg)

Console: Capturing Image… (5sec pause)

The ground target image has been captured successfully.

\*\*Console: Choose from the following menu of commands:

1. Status Checks
2. Slew Image
3. Acquire Target
4. Capture Image
5. Transfer Telemetry

**$ 5** {function name: def transferTelemetry)

Console: Transferring Payload Telemetry… (5sec pause)

Data has been Transferred! GREAT WORK ON THE PAYLOAD SYSTEM CONSOLE!

====================================================

**Communications script (from Comms.py- DONE):**

(\*\* = repeated command)

(Note: Subsystem has no user input)

Console: WELCOME TO THE COMMUNICATIONS (COMMs) CONSOLE!

Your task is to verify that signal lock is established between the Ku-Band satellite antenna and the ground station antenna, transmit the target image to the ground station, process the image, and display the results.

\*\*Console: Choose from the following menu of commands:

1. Status Checks
2. Verify Signal
3. Signal Gain
4. Signal Frequency
5. Download Telemetry Data
6. Process Telemetry Data
7. Display Image

**$ 1** {function name: def systemChecks)

Console: Checking Power Systems… (5sec pause)

The SimCraft’s current On-Board Computer Status is [Reached/Not Reached (true/false)]

The SimCraft’s current Antenna Status is [Reached/Not Reached (true/false)]

If an error occurs, run the Refresh command to refresh and update the system.

{Note (Summer): Make sure they can all end up true}

\*\*Console: Choose from the following menu of commands:

1. Status Checks
2. Verify Signal
3. Signal Gain
4. Signal Frequency
5. Download Telemetry Data
6. Process Telemetry Data
7. Display Image

**$ 2** {function name: def verifySignal)

Console: Verifying Signal… (5sec pause)

The SimCraft’s current Signal Frequency is [Inside/Outside] the required Bandwidth of 12.000-18.000 GHz.

The SimCraft’s current Signal Gain is [Inside/Outside] the required strength of 25-30 dB.

{Note (Summer): Make sure they can all end up true}

\*\*Console: Choose from the following menu of commands:

1. Status Checks
2. Verify Signal
3. Signal Gain
4. Signal Frequency
5. Download Telemetry Data
6. Process Telemetry Data
7. Display Image

**$ 3** {function name: def signalGain)

Console: How much do you want to change the Signal Gain (in dB)?

**$ *#*** {input var/type: newGain/int} (5sec pause)

Console: The SimCraft’s Signal Gain has changed by # dB.

**$ 4** {function name: def signalFreq)

Console: How much do you want to change the Signal Frequency (in GHz)?

**$ *#*** {input var/type: newFreq/int} (5sec pause)

Console: The SimCraft’s Signal Frequency has changed by # dB.

\*\*Console: Choose from the following menu of commands:

1. Status Checks
2. Verify Signal
3. Signal Gain
4. Signal Frequency
5. Download Telemetry Data
6. Process Telemetry Data
7. Display Image

**$ 5** {function name: def downloadTelemetry)

Console: Downloading Subsystem Telemetry… (5sec pause)

ACS Telemetry…. Complete!

EPS Telemetry…. Incomplete!

TCS Telemetry…. Complete!

Payload Telemetry…. Complete!

The Subsystem Telemetry Data has been successfully downloaded!

**OR**

Some Subsystems have not completed their missions yet and need to send their Telemetry Data to finish your task.

\*\*Console: Choose from the following menu of commands:

1. Status Checks
2. Verify Signal
3. Signal Gain
4. Signal Frequency
5. Download Telemetry Data
6. Process Telemetry Data
7. Display Image

**$ 6** {function name: def processTelemetry)

Console: Processing SimCraft Telemetry… (5sec pause)

The Subsystem Telemetry Data has been successfully Processed!

**OR**

Some Subsystems have not completed their missions yet and need to send their Telemetry Data to finish your task.

\*\*Console: Choose from the following menu of commands:

1. Status Checks
2. Verify Signal
3. Signal Gain
4. Signal Frequency
5. Download Telemetry Data
6. Process Telemetry Data
7. Display Image

**$ 7** {function name: def displayImg)

Console: Displaying Image… (5sec pause)

Some Subsystems have not completed their missions yet and need to send their Telemetry Data to finish your task.

**OR**

The Subsystem Telemetry Data has been successfully Processed!

Click the link to view the image! {url -> **rick roll**}

GREAT WORK ON THE PAYLOAD SYSTEM CONSOLE!

Mission Accomplished! (5sec pause)

Just Kidding… Here’s the actual Image {url -> img on new page}

====================================================