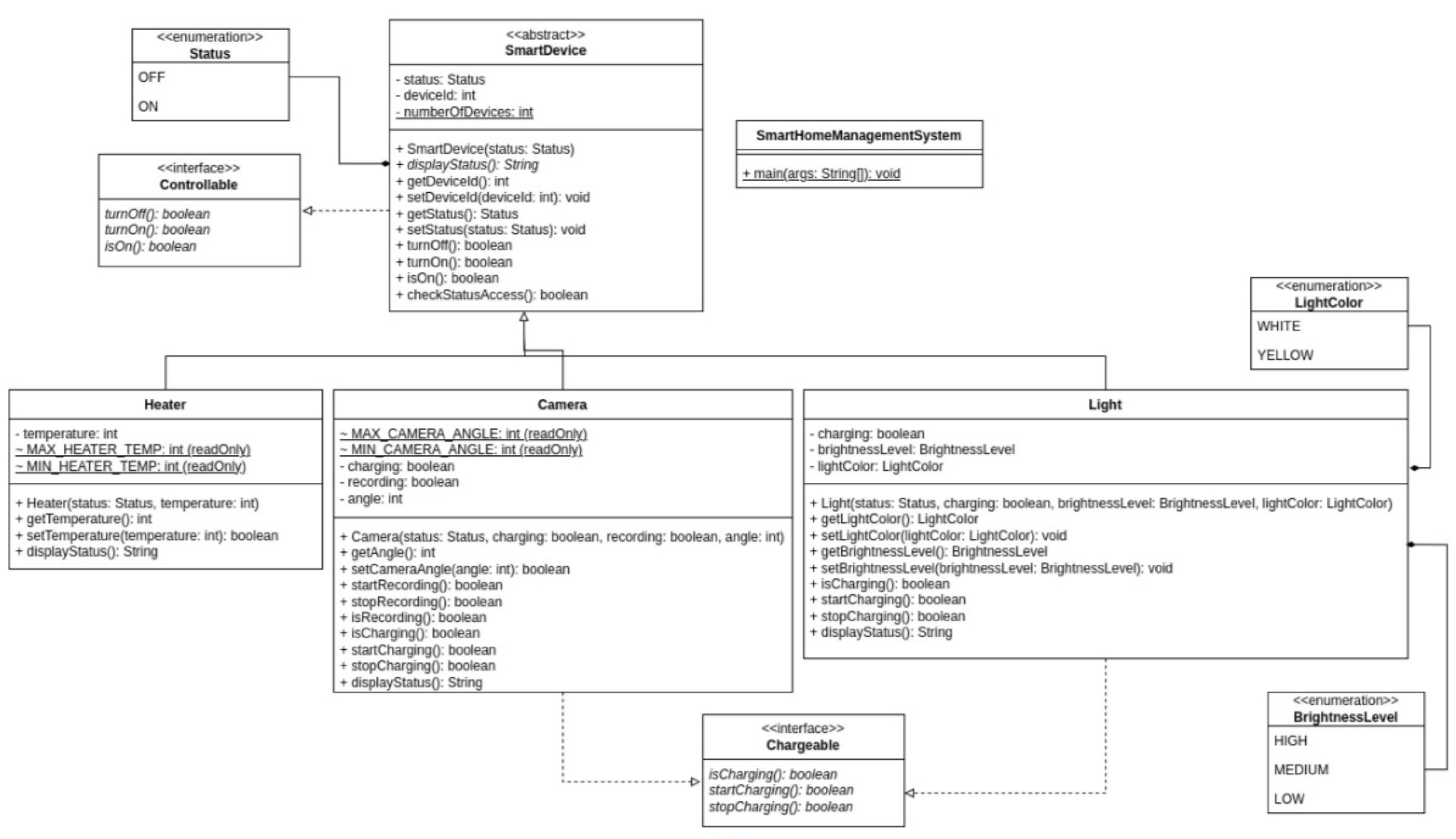
A. Smart Home system

time limit per test: 1 second
memory limit per test: 256 megabytes

Smart Home System

Description:

The smart home system is designed to manage various smart devices, including Lights, Cameras, and Heaters. Each device has capabilities for being activated or deactivated, alongside particular features and settings that can be manipulated when the devices are in the ON state. Here is the UML class diagram that represents the structure of the code:


<<enumeration>>
Status
OFF
ON
<<interface>>
Controllable
turnOff(): boolean
turnOn(): boolean
isOn(): boolean
<<abstract>>
SmartDevice
- status: Status
- deviceId: int
- numberOfDevices: int
+ SmartDevice(status: Status)
+ displayStatus(): String
+ getDeviceId(): int
+ setDeviceId(deviceId: int): void
+ getStatus(): Status
+ setStatus(status: Status): void
+ turnOff(): boolean
+ turnOn(): boolean
+ isOn(): boolean
+ checkStatusAccess(): boolean
SmartHomeManagementSystem
+ main(args: String[]): void
<<enumeration>>
LightColor
WHITE
YELLOW
Heater
- temperature: int
~ MAX_HEATER_TEMP: int (readOnly)
~ MIN_HEATER_TEMP: int (readOnly)
+ Heater(status: Status, temperature: int)
+ getTemperature(): int
+ setTemperature(temperature: int): boolean
+ displayStatus(): String
Camera
~ MAX_CAMERA_ANGLE: int (readOnly)
~ MIN_CAMERA_ANGLE: int (readOnly)
- charging: boolean
- recording: boolean
- angle: int
+ Camera(status: Status, charging: boolean, recording: boolean, angle: int)
+ getAngle(): int
+ setCameraAngle(angle: int): boolean
+ startRecording(): boolean
+ stopRecording(): boolean
+ isRecording(): boolean
+ isCharging(): boolean
+ startCharging(): boolean
+ stopCharging(): boolean
+ displayStatus(): String
Light
- charging: boolean
- brightnessLevel: BrightnessLevel
- lightColor: LightColor
+ Light(status: Status, charging: boolean, brightnessLevel: BrightnessLevel, lightColor: LightColor)
+ getLightColor(): LightColor
+ setLightColor(lightColor: LightColor): void
+ getBrightnessLevel(): BrightnessLevel
+ setBrightnessLevel(brightnessLevel: BrightnessLevel): void
+ isCharging(): boolean
+ startCharging(): boolean
+ stopCharging(): boolean
+ displayStatus(): String
<<interface>>
Chargeable
isCharging(): boolean
startCharging(): boolean
stopCharging(): boolean
<<enumeration>>
BrightnessLevel
HIGH
MEDIUM
LOW

UML

- **Device States:**
  - **Lights:** Each light can be in one of two states: ON or OFF. When the lights are ON, their brightness and color can be modified. When they are OFF, these attributes cannot be changed.
  - **Cameras:** Each camera can also be in one of two states: ON or OFF. When ON, cameras can record, and their angles can be adjusted. When OFF, cameras cannot record, and their angles cannot be changed.
  - **Heaters:** Each heater has two states: ON or OFF. When ON, the temperature can be set within the specified range. When OFF, the temperature cannot be modified.
- **Initial Status of the Home:**
  - Lights: There are four lights, all of which are initially turned ON. They have a LOW brightness setting and are configured to emit a YELLOW color. They are not charging.
  - Cameras: The system includes two cameras, both initially turned ON. They are not recording, not charging, and are positioned with a camera angle of 45°.
  - Heaters: The system has four heaters, all turned ON and set to a temperature of 20°C.

Rules:

- When any device is turned OFF, adjustments to any attributes are restricted except the charging status attribute for Lights and Cameras.
- The lights can only be set to either YELLOW or WHITE colors.
- Brightness levels can be categorized as LOW, MEDIUM, or HIGH.
- Each camera's angle can be adjusted independently within a vertical integer range from -60° to +60°.
- The temperature settings for heaters must remain within the integer range from 15°C to 30°C.
- The Lights and Cameras are chargeable, while the Heaters are not.

Initial Device IDs and Names:

- Lights: The lights are identified by device IDs ranging from 0 to 3.
- Cameras: The cameras are identified by device IDs 4 and 5.
- Heaters: The heaters are identified by device IDs ranging from 6 to 9.
- Names: The name of any device should be from the following set of names: {Camera, Heater, Light}.

Commands:

Command	Description
DisplayAllStatus	Displays the current status of all devices in the system.
TurnOn {*deviceName*} {*deviceId*}	Turns a specified device ON.
TurnOff {*deviceName*} {*deviceId*}	Turns a specified device OFF.
StartCharging {*deviceName*} {*deviceId*}	Starts charging a specified device, if chargeable.
StopCharging {*deviceName*} {*deviceId*}	Stops charging a specified device, if chargeable.
SetTemperature {*deviceName*} {*deviceId*} {*temperature*}	Sets the temperature for a specified heater device within its allowable range [15, 30].
SetBrightness {*deviceName*} {*deviceId*} {*brightnessLevel*}	Sets the brightness level (LOW, MEDIUM, HIGH) for a specified light device.
SetColor {*deviceName*} {*deviceId*} {*color*}	Sets the color (YELLOW or WHITE) for a specified light device.
SetAngle {*deviceName*} {*deviceId*} {*angle*}	Adjusts the camera angle within the range [-60, 60] for a specified camera device.
StartRecording {*deviceName*} {*deviceId*}	Starts recording for a specified camera device.
StopRecording {*deviceName*} {*deviceId*}	Stops recording for a specified camera device.
end	It is the end of the input, after which the program should stop.

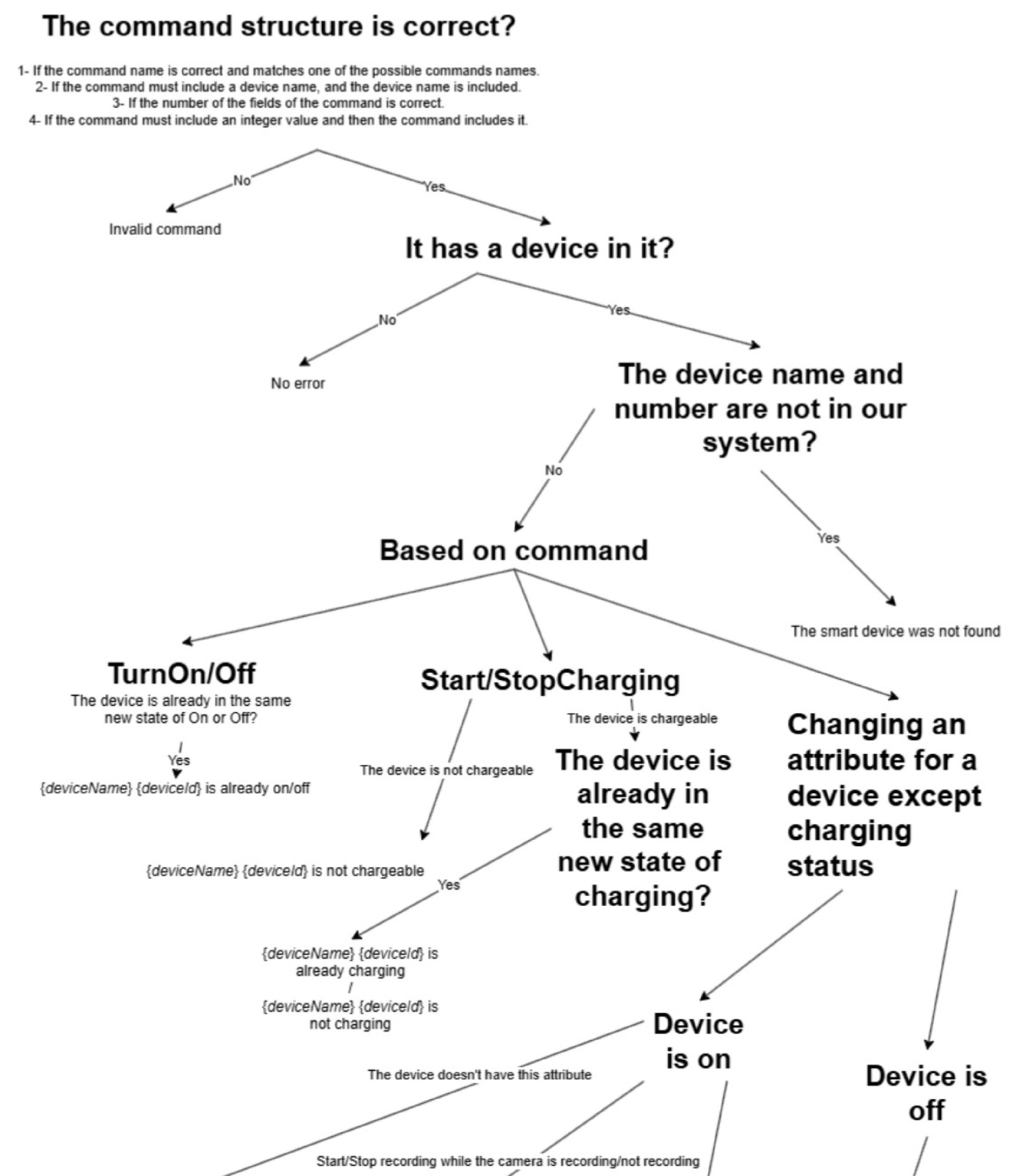
The **DisplayAllStatus** command displays the current status of each device in the system by calling the `displayStatus` method for every device in an array of SmartDevice objects.

Each type of device has its own `displayStatus` method that returns a formatted string with the device's status information, which is printed to the console. Here is how each type of device's status is displayed:

- For Light devices:
  - Returns information about its ID, power status (ON or OFF), color, charging status, and brightness level.
  - Example output format: "Light *{deviceId}* is *{status}*, the color is *{color}*, the charging status is *{chargingStatus}*, and the brightness level is *{brightnessLevel}*."
- For Camera devices:
  - Returns information about its ID, power status, angle, charging status, and recording status.
  - Example output format: "Camera *{deviceId}* is *{status}*, the angle is *{angle}*, the charging status is *{chargingStatus}*, and the recording status is *{recordingStatus}*."
- For Heater devices:
  - Returns information about its ID, power status, and temperature.
  - Example output format: "Heater *{deviceId}* is *{status}* and the temperature is *{temperature}*."

Errors:

Condition	Error Message
The command doesn't follow the structure of a command or has incorrect values	Invalid command
If a device with the specified name and ID was not found	The smart device was not found
Charging a non-chargeable device	{*deviceName*} {*deviceId*} is not chargeable
Stopping charging on a non-chargeable device	{*deviceName*} {*deviceId*} is not chargeable
Setting temperature for a non-heater device	{*deviceName*} {*deviceId*} is not a heater
Setting brightness level out of scope	The brightness can only be one of "LOW", "MEDIUM", or "HIGH"
Setting brightness level for a non-light device	{*deviceName*} {*deviceId*} is not a light
Setting color of a light to an invalid value	The light color can only be "YELLOW" or "WHITE"
Setting color for a non-light device	{*deviceName*} {*deviceId*} is not a light
Setting camera angle for a non-camera device	{*deviceName*} {*deviceId*} is not a camera
Starting to record on a non-camera device	{*deviceName*} {*deviceId*} is not a camera
Stopping to record on a non-camera device	{*deviceName*} {*deviceId*} is not a camera
Turning off a device that is already off	{*deviceName*} {*deviceId*} is already off
Turning on a device that is already on	{*deviceName*} {*deviceId*} is already on
Changing the device's attributes while it is off (except charging state)	You can't change the status of the {*deviceName*} {*deviceId*} while it is off
Setting camera angle out of allowed range	Camera {*deviceId*} angle should be in the range [-60, 60]
Starting to record when the camera is already recording	{*deviceName*} {*deviceId*} is already recording
Stopping to record when the camera is not recording	{*deviceName*} {*deviceId*} is not recording
Charging a device that is already charging	{*deviceName*} {*deviceId*} is already charging
Stopping charging on a device that is not charging	{*deviceName*} {*deviceId*} is not charging
Setting heater temperature out of range	Heater {*deviceId*} temperature should be in the range [15, 30]


The command structure is correct?
1- If the command name is correct and matches one of the possible commands names.
2- If the command must include a device name, and the device name is included.
3- If the number of the fields of the command is correct.
4- If the command must include an integer value and then the command includes it.
No
Yes
Invalid command
It has a device in it?
No
Yes
No error
The device name and number are not in our system?
No
Yes
Based on command
The smart device was not found
TurnOn/Off
The device is already in the same new state of On or Off?
Yes
{deviceName} {deviceId} is already on/off
Start/StopCharging
The device is not chargeable
{deviceName} {deviceId} is not chargeable
The device is chargeable
The device is already in the same new state of charging?
Yes
{deviceName} {deviceId} is already charging
/
{deviceName} {deviceId} is not charging
Changing an attribute for a device except charging status
Device is on
Device is off
The device doesn't have this attribute
{deviceName} {deviceId} is not a light/ camera/ heater
Start/Stop recording while the camera is recording/not recording
{deviceName} {deviceId} is already recording
{deviceName} {deviceId} is not recording
Value out of range
The brightness can only be one of "LOW", "MEDIUM", or "HIGH"
The light color can only be "YELLOW" or "WHITE"
Camera {deviceId} angle should be in the range [-60, 60]
Heater {deviceId} temperature should be in the range [15, 30]
You can't change the status of the {deviceName} {deviceId} while it is of

Errors order

Successful Outputs:

Condition	Success Message
TurnOn	{*deviceName*} {*deviceId*} is on
TurnOff	{*deviceName*} {*deviceId*} is off
StartCharging	{*deviceName*} {*deviceId*} is charging
StopCharging	{*deviceName*} {*deviceId*} stopped charging
SetTemperature	{*deviceName*} {*deviceId*} temperature is set to {*temperature*}
SetBrightness	{*deviceName*} {*deviceId*} brightness level is set to {*brightness*}
SetColor	{*deviceName*} {*deviceId*} color is set to {*color*}
SetAngle	{*deviceName*} {*deviceId*} angle is set to {*angle*}
StartRecording	{*deviceName*} {*deviceId*} started recording
StopRecording	{*deviceName*} {*deviceId*} stopped recording

Input

The input consists of an arbitrary number of lines at least one containing "end". Each test contains a sequence of commands in of following types:

- `DisplayAllStatus`
- `TurnOn` {deviceName} {deviceId}
- `TurnOff` {deviceName} {deviceId}
- `StartCharging` {deviceName} {deviceId}
- `StopCharging` {deviceName} {deviceId}
- `SetTemperature` {deviceName} {deviceId} {temperature}
- `SetBrightness` {deviceName} {deviceId} {brightnessLevel}
- `SetColor` {deviceName} {deviceId} {color}
- `SetAngle` {deviceName} {deviceId} {angle}
- `StartRecording` {deviceName} {deviceId}
- `StopRecording` {deviceName} {deviceId}
- `end`

The order of commands may vary based on the state of the devices and may generate different outputs or errors based on the commands' validity and the current state of the devices. The program continues the execution while the end command is not met.

Output

For each command output the result is based on the operation executed. If the command is valid and executed successfully, provide the status per the rules defined in the problem statement. If the command fails due to an error condition, output the appropriate error message as specified in a different line.

Examples

input Скопировать

```
TurnOn Camera 5
TurnOn Camera 6
end
```

output Скопировать

```
Camera 5 is already on
The smart device was not found
```

input Скопировать

```
kll
end
```

output Скопировать

```
Invalid command
```