Function: thumbNail(img: Image):file

Parameter: Takes an image to shrink or expand an image to fit a set image size.

- one Parameter:

- Image img

- Characteristics of interest:

- File format

- image width

- image height

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Parameters | Characteristics | Blocks | Values | Constraints | Label |

Image img File format valid formats [.bmp,.gif,.jpg,.png,.tif,.tiff] [single] B1

!valid formats other [error] B2

Image width empty [0] [error] B3

!empty [1, max] [h !empty] B4

Image height empty [0] [error] B5

!empty [1, max] [w !empty] B6

All combination coverage

|  |  |
| --- | --- |
| Test Frames | Test Cases |
| B1 (or B1, B4, B6) | Test.jpg |
| B2 | Test.pdf |
| B3 | none |
| B5 | none |

Function: calcHori(width: int, height int): int

Specification: Calculates the number of pixels horizontally by dividing by the ratio that reduced the height to 48 pixels.

Special test frame: The number of horizontal pixels must have an upper bound to prevent unreasonably wide images and to allow the Arduino to signal at a leisurely rate. An arbitrary number of 250 chosen, for 20 micro second intervals.

- Two parameters:

- int w, int h

- Characteristics of parameter w:

- Value

- Characteristics of parameter h:

- Value

- Special test frame:

- A width and height that causes the calculated horizontal pixels to be >= 250.

i.e. (500, 3000) [single] B7

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Parameters | Characteristics | Blocks | Values | Constraints | Label |

Integer width, w Value Minimum(1) (-∞, 1) [error] B1

Nominal(500) [1, 1920] B2

(*soft constraint)* Maximum(1920) (1920, ∞) [single] B3

Integer height, h Value Minimum(1) (-∞, 1) [error] B4

Nominal(500) [1, 1920] B5

(*soft constraint)* Maximum(1920) (1920, ∞) [single] B6

All Combinations coverage

|  |  |
| --- | --- |
| Test Frames | Test Cases |
| B1 | W=-5, h =10 |
| B2, B5 | W=100, h = 200 |
| B3 (or B3, B5) | W= 3000, h= 1000 |
| B4 | W = 48, h = 0 |
| B6 (or B6, B2) | W= 1500, h = 4000 |
| B7 | W = 500, h = 3000 |

Function: black\_and\_white(img: Image)

Specification: Converts an image to black and white.

- one Parameter:

- Image img

- Characteristics of interest:

- File format

- image width

- image height

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Parameters | Characteristics | Blocks | Values | Constraints | Label |

Image img File format valid formats [.bmp,.gif,.jpg,.png,.tif,.tiff] [single] B1

!valid formats other [error] B2

Image width empty [0] [error] B3

!empty [1, max] [h !empty] B4

Image height empty [0] [error] B5

!empty [1, max] [w !empty] B6

All combination coverage

|  |  |
| --- | --- |
| Test Frames | Test Cases |
| B1 (or B1, B4, B6) | Test.jpg |
| B2 | Test.pdf |
| B3 | none |
| B5 | none |

Function: bitArray(img: Image): bitMatrix

Specification: Takes the black and white image and fills a 2d array with 0, for black, and 1, for white.

- one Parameter:

- Image img

- Characteristics of interest:

- File format

- image width

- image height

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Parameters | Characteristics | Blocks | Values | Constraints | Label |

Image img File format valid formats [.bmp,.gif,.jpg,.png,.tif,.tiff] [single] B1

!valid formats other [error] B2

Image width empty [0] [error] B3

!empty [1, max] [h !empty] B4

Image height empty [0] [error] B5

!empty [1, max] [w !empty] B6

All combination coverage

|  |  |
| --- | --- |
| Test Frames | Test Cases |
| B1 (or B1, B4, B6) | Test.jpg |
| B2 | Test.pdf |
| B3 | none |
| B5 | none |

Function: printBitArray(matrix: 2d array)

Specification: Print the 2d array that represents the bit pattern for a black and white image.

- One parameter:

- 2d array “matrix”

- Characteristics:

- array size, rows and cols

- element values

- type of elements

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Parameters | Characteristics | Blocks | Values | Constraints | Label |

2d array, matrix array size, row empty [0] [error] B1

Not empty [1, 48] [col !empty] B2

> max (48, ∞) [single] B3

array size, col empty [0] [error] B4

Not empty [1, 250] [row !empty] B5

> max (250, ∞) [single] B6

Element values zero [0] B7

one [1] B8

not valid [!0 or !1] [single] B9

type of elements integer int B10

not int !int [single] B11

All combination coverage

|  |  |
| --- | --- |
| Test Frames | Test Cases |
| B1 | Empty matrix |
| B2, B5, B7, B10 | [0][0]  [0][0] |
| B2, B5, B8, B10 | [1][1][1][1]  [1][1][1][1] |
| B3 (or B3, B5, B7, B10) | [0][0]  [0][0]  . .  . . 49 rows |
| B4 | Empty matrix |
| B6 (or B2, B6, B8, B10) | [1][1] . . 251 cols |
| B9 (or B2, B5, B9, B10) | [2][2]  [3][4] |
| B11 (or B2,B5, B9, B11) | [h][i] |

Function: signalInterval(width: Int): float

Specification: Calculates the interval to signal the LEDs for the next bit pattern. An arbitrary number of 250 chosen, for 20 micro second intervals, as the upper bound.

- One parameter:

- int w

- Characteristics of parameter w:

- Value

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Parameters | Characteristics | Blocks | Values | Constraints | Label |

Integer width, w Value Minimum(1) (-∞, 1) [error] B1

Nominal(50) [1, 250) B2

(*arbitrary constraint)* Maximum(150) [250, ∞) [single] B3

All combination coverage

|  |  |
| --- | --- |
| Test Frames | Test Cases |
| B1 | W= -5 |
| B2 | W=50 |
| B3 | W=300 |