LUMINANCE

1 This algorithm calculates the luminance of a pixel.

```
Algorithm 1 Luminance
Require:
    A colour defined as a tuple of integers in 8-bit RGB format such that:
    $0 \leq c_{0..2} \leq 255$
Ensure:
    The luminance:
    L

    $L \leftarrow \frac{\sum_{i=0}^{2} c_i}{3}$
```

COLOUR TOLERANCE

1 This algorithm checks if an existing pixel is close to another in colour

```
Algorithm 1 Colour Tolerance
Require:
    a threshold value, $0 \leq t \leq 255$
    a colour in RGB format, $0 \leq c_{0..2} \leq 255$
    a pixel in RGB format, $0 \leq p_{0..2} \leq 255$
function TOLERANCE(color c, pixel p, threshold t)
    $d \leftarrow \sum_{i=0}^{2}(p_i - c_i)^2$
    if $r < t$ then
        return true
    else
        return false
    end if
end function
```

COLOUR DISTANCE

This algorithm can be used to determine the 'distance' between two colours. It is fundamental to a range of other algorithms in media computation.

```
Algorithm 1 Calculate Distance Between Two Colours
Require:
    Two colours defined as a tuple of integers in 8-bit RGB format such that:
    $0 \leq r_{0..1} \leq 255$
    $0 \leq g_{0..1} \leq 255$
    $0 \leq b_{0..1} \leq 255$
Ensure:
    The distance between the two colours:
    d

$d \leftarrow \sqrt{(r_1 - r_0)^2 + (g_1 - g_0)^2 + (b_1 - b_0)^2}$
return $d$
```

POSTERIZATION

1 This algorithm reduces the colours within an image. Multiple conditions might be required.

Algorithm 1 Posterization

```
Require:
    a channel value, $0 \leq c_{0..2} \leq 255$
    a replacement value, $0 \leq r \leq 255$
    a minimum threshold, $0 \leq t_{min} \leq 255$
    a maximum threshold, $0 \leq t_{max} \leq 255$
procedure POSTERIZATION(t, c, r)
    for $x = 0$, width, $y = 0$,height do
        $c \leftarrow$ pixel(x, y)
        if $t_{min} \leq c_0 \leq t_{max}$ then
            $c_0 \leftarrow r$
            pixel(x, y) $\leftarrow c$
        end if
    end for
end procedure
```