```
// Returns the Red component of a 32-bit color
int GetRed (uint32_t tmpColor)
{
    return (int) ( (tmpColor >> 16) & 0xFF );
}

// Returns the Green component of a 32-bit color
int GetGreen (uint32_t tmpColor)
{
    return (int) ( (tmpColor >> 8) & 0xFF );
}

// Returns the Blue component of a 32-bit color
int GetBlue (uint32_t tmpColor)
{
    return (int) ( tmpColor & 0xFF );
}


// Returns a 32-bit color built out of Red, Green, and Blue elements
uint32_t BuildColor (int red, int green, int blue)
{
    return ( (uint32_t) (red << 16) ) | ( (uint32_t) (green << 8) ) | ( (uint32_t) blue );
}


// Returns a 32-bit color generated as a cross-fade between a start color and an end color
uint32_t CrossFadeColor (uint32_t startColor, uint32_t endColor, int numSteps, int currentStep)
{
    uint32_t tmpRed;
    uint32_t tmpGreen;
    uint32_t tmpBlue;

    tmpRed =   ( (  GetRed(startColor) * (numSteps - currentStep) ) + (  GetRed(endColor) * currentStep) ) / numSteps;
    tmpGreen = ( (GetGreen(startColor) * (numSteps - currentStep) ) + (GetGreen(endColor) * currentStep) ) / numSteps;
    tmpBlue =  ( ( GetBlue(startColor) * (numSteps - currentStep) ) + ( GetBlue(endColor) * currentStep) ) / numSteps;

    return BuildColor( ((int) tmpRed), ((int) tmpGreen), ((int) tmpBlue) );
}
```