Question 1

How many unique colour values can the colour value integer contain?

The colour is stored in a 32-bit unsigned integer.

✅ Answer: 4,294,967,296 unique values.

Question 2

What is the minimum value, maximum value, and range for each colour component?

Each component (R, G, B, A) is 8 bits (1 byte).

Minimum = 0

Maximum = 255

Range = 256

✅ Answer: Min = 0, Max = 255, Range = 256.

Question 3

Write decimal 94 as an 8-bit binary number.

94 in binary = 01011110.

✅ Answer: 01011110

Question 4

Store red = 94 into the left-most byte (R) of RGBA, all others = 0.

R = 01011110
G = 00000000
B = 00000000
A = 00000000

Combined 32-bit value = 01011110 00000000 00000000 00000000

Answer: 01011110000000000000000000000000

Question 5

Decimal value of the binary from Question 4.

That’s 94 x 2^24 = 1,577,058,304

✅ Answer: 1,577,058,304

Question 6

C++ bit shifting operation to move R → G.

unsigned int val = colour;
val = (val >> 8) & 0x00FF0000;

This moves bits from the red position (24–31) down into green’s position (16–23).

✅ Answer: val = ((colour >> 8) & 0x00FF0000);

Question 7

Now only the green component is set to 94.

Binary:
00000000 01011110 00000000 00000000

Decimal: 94 x 2^16 = 6,164,480

✅ Answer: 6,164,480

Binary: 00000000010111100000000000000000

Decimal: 6,164,480

Question 8
Colour.h

```
#pragma once
using Byte = unsigned char;

class Colour
{
private:
   unsigned int colour; // 32-bit RGBA

public:
   Colour() : colour(0) {}

   Colour(Byte red, Byte green, Byte blue, Byte alpha) {
      setRed(red);
      setGreen(green);
      setBlue(blue);
      setAlpha(alpha);
   }
Byte getRed()   const { return (colour >> 24) & 0xFF; }
   Byte getGreen() const { return (colour >> 16) & 0xFF; }
   Byte getBlue()  const { return (colour >> 8)  & 0xFF; }
   Byte getAlpha() const { return  colour        & 0xFF; }

   void setRed(Byte red) {
      colour &= 0x00FFFFFF;
      colour |= (red << 24);
   }
   void setGreen(Byte green) {
      colour &= 0xFF00FFFF;
      colour |= (green << 16);
   }
   void setBlue(Byte blue) {
      colour &= 0xFFFF00FF;
      colour |= (blue << 8);
   }
   void setAlpha(Byte alpha) {
      colour &= 0xFFFFFF00;
      colour |= alpha;
   }
   unsigned int getValue() const { return colour; }
   void setValue(unsigned int value) { colour = value; }
};
```