1a) index 1 = segment 1, index 2 = segment 2, index 3 = segment 3

Gradient ((y2 – y1) / (x2 – x1))
M1 = (40 – 0) / (100 – 0) = 0.4
M2 = (200 – 40) / (165 – 100) = 2.4615
M3 = (255 – 200) / (255 – 165) = 0.6111
Intercept point (y = mx + c)
C1
y = 0.4x + c
40 = 0.4(100) + c
40 = 40 + c
c = 0
therefore 1st function, y = 0.4x
C2
y = 2.4615x + c
200 = 2.4615 (165) + c
200 = 406.1475 + c
c = -206.1475
therefore 2nd function, y = 2.4615x – 206.1475
C3
y = 0.6111x + c
255 = 0.6111 (255) + c
255 = 155.8305 + c
c = 99.1695
therefore 3rd function, y = 0.6111x + 99.1695

$$f(x) = \begin{cases} 0.4x & \text{if } 0 \leq x < 100 \\ 2.4615x - 206.1475 & \text{if } 100 \leq x < 165 \\ 0.6111x + 99.1695 & \text{if } 165 \leq x \leq 255 \end{cases}$$

b)
Before rounding

2.4615(108) – 206.1475 = 59.6945	0.6111(190) + 99.1695 = 215.2785	0.6111(255) + 99.1695 = 255
0.4(44) = 17.6	0.4(82) = 32.8	0.6111(249) + 99.1695 = 251.3334
2.4615(125) – 206.1475 = 101.54	2.4615(163) – 206.1475 = 195.077	0.4(0) = 0

Output image patch (rounding)

60	215	255
18	33	251
102	195	0

c) The values are stretched based on the input values and which transform function is applied to the input, like example values such as 163 increased to 195. It stretches out the input giving more different range of values.
Lower input values < 100 becomes lower (darker) because first function lowers down the input values.
Middle input values >= 100 and < 165 changes depending on the input value, the closer it is to 165 the output will be higher(stretched out).
Higher input values >= 165 and <= 255 will amplify the value (example input 165, output = 200).

2.

a)

Fk	Count	PF(I)	Cumulative PF(I)	$g(l)=\left\lfloor\left(\sum_{k=0}^{l} p_F(k)\right)*(L-1)\right\rfloor$ Where (8-1) = 7	PG(I)
0	2	.08	.08	[0.56] = 1	0
1	6	.24	.32	[2.24] = 2	.08
2	2	.08	.40	[2.80] = 3	.24
3	1	.04	.44	[3.08] = 3	.08 + .04 = .12
4	2	.08	.52	[3.64] = 4	.08 + .12 = .2
5	3	.12	.64	[4.48] = 4	0
6	6	.24	.88	[6.16] = 6	0.24
7	3	.12	1.00	[7.00] = 7	0.12

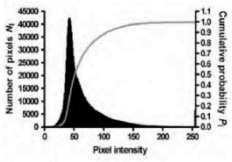
Output 3bit image after histogram equalization

1	3	3	6	2
6	4	7	6	2
4	6	2	7	4
2	4	1	7	6
2	6	3	2	4

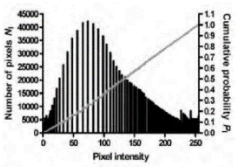
b) i) The values before histogram was applied is lower, as compared to values after histogram was applied. Therefore, the image's color intensity are intensified.
ii) The histogram becomes more flatter, the color intensity with of lower frequency such as 0 and 5 are intensified to other colors to balance(equalize) out the histogram.

c) yes, the image pixel values will change further if histogram equalization is applied over and over again until the point where the histogram becomes almost completely flat(even), then no changes will be made because it's already flat.

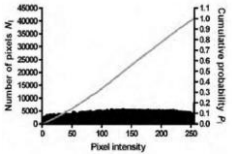
Some sample of iterative histogram equalization as it becomes flatter:
1st iteration



2nd iteration



3rd iteration



3.
256 colors can be represented in 2^8, therefore 8 bits required to store all 256 colors.
Single channel = 8 bits = 1 bytes

Standard full color
RGB Channel = 3 * 8 bits = 3 bytes
Image storage size = M * N * 3 * 8 bits = M * N * 3 bytes

Indexed color
Palette size = 256 * 3 * 8 bits = 678bytes
Image Storage size = (8 * M * N) + (256 * 3 * 8 bits) = (M * N) + (768) bytes

Storage saved = standard full color storage size – indexed color storage size
= (M * N * 3) – ((M * N) + (768)) bytes

If the image size is smaller then standard full color will save more space for example:
M = 6, N = 6
Standard full color = 6 * 6 * 3 = 108 bytes
Where
Indexed color = 6 * 6 + 768 = 804 bytes

But if the image size is very large then indexed color will save more space for example:
M = 1024, N = 1024
Standard full color = 1024 * 1024 * 3 = 3145728 bytes
Where
Indexed color = 1024 * 1024 + 768 = 1049344 bytes