1. A) i)

Left grid (columns A–I):

	A	B	C	D	E	F	G	H	I
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	1	1	0	0	0
4	0	0	1	1	1	1	1	0	0
5	0	0	0	1	1	1	1	0	0
6	0	0	1	1	1	1	0	0	0
7	0	0	0	0	1	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0

Right grid (columns K–S):

	K	L	M	N	O	P	Q	R	S
1	0	0	0	0	0	0	0	0	0
2	0	0	0	1	1	1	1	0	0
3	0	1	1	1	1	1	1	1	0
4	0	1	1	1	1	1	1	1	0
5	0	1	1	1	1	1	1	1	0
6	0	1	1	1	1	1	1	1	0
7	0	1	1	1	1	1	1	0	0
8	0	0	0	1	1	1	0	0	0
9	0	0	0	0	0	0	0	0	0

ii)

Left grid (columns A–I):

	A	B	C	D	E	F	G	H	I
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	1	1	0	0	0
4	0	0	1	1	1	1	1	0	0
5	0	0	0	1	1	1	1	0	0
6	0	0	1	1	1	1	0	0	0
7	0	0	0	0	1	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0

Right grid (columns K–S):

	K	L	M	N	O	P	Q	R	S
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0
5	0	0	0	0	1	0	0	0	0
6	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0

iii)

Left grid (columns A–I):

	A	B	C	D	E	F	G	H	I
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	1	1	0	0	0
4	0	0	1	1	1	1	1	0	0
5	0	0	0	1	1	1	1	0	0
6	0	0	1	1	1	1	0	0	0
7	0	0	0	0	1	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0

Right grid (columns K–S):

	K	L	M	N	O	P	Q	R	S
1	0	0	0	0	0	0	0	0	0
2	0	0	0	1	1	1	1	0	0
3	0	1	1	1	1	1	1	1	0
4	0	1	1	1	1	1	1	1	0
5	0	1	1	1	1	1	1	1	0
6	0	1	1	1	1	1	1	1	0
7	0	1	1	1	1	1	1	0	0
8	0	0	0	1	1	1	0	0	0
9	0	0	0	0	0	0	0	0	0
10									
11	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0
13	0	0	0	0	1	1	0	0	0
14	0	0	1	1	1	1	1	0	0
15	0	0	1	1	1	1	1	0	0
16	0	0	1	1	1	1	0	0	0
17	0	0	0	0	1	0	0	0	0
18	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0

iv)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0
3	0	0	0	0	1	1	0	0	0		0	0	0	0	0	0	0	0	0
4	0	0	1	1	1	1	1	0	0		0	0	0	0	0	0	0	0	0
5	0	0	0	1	1	1	1	0	0		0	0	0	0	1	0	0	0	0
6	0	0	1	1	1	1	0	0	0		0	0	0	0	0	0	0	0	0
7	0	0	0	0	1	0	0	0	0		0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0
10																			
11											0	0	0	0	0	0	0	0	0
12											0	0	0	0	0	0	0	0	0
13											0	0	0	0	0	0	0	0	0
14											0	0	0	1	1	1	0	0	0
15											0	0	0	1	1	1	0	0	0
16											0	0	0	1	1	1	0	0	0
17											0	0	0	0	0	0	0	0	0
18											0	0	0	0	0	0	0	0	0
19											0	0	0	0	0	0	0	0	0

b) i)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0		0	0	0	0	1	1	0	0	0
3	0	0	0	0	1	1	0	0	0		0	0	1	1	1	1	1	0	0
4	0	0	1	1	1	1	1	0	0		0	1	1	1	1	1	1	1	0
5	0	0	0	1	1	1	1	0	0		0	0	1	1	1	1	1	1	0
6	0	0	1	1	1	1	0	0	0		0	1	1	1	1	1	1	0	0
7	0	0	0	0	1	0	0	0	0		0	0	1	1	1	1	0	0	0
8	0	0	0	0	0	0	0	0	0		0	0	0	0	1	0	0	0	0
9	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0

ii)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0
3	0	0	0	0	1	1	0	0	0		0	0	0	0	0	0	0	0	0
4	0	0	1	1	1	1	1	0	0		0	0	0	0	1	1	0	0	0
5	0	0	0	1	1	1	1	0	0		0	0	0	0	1	1	0	0	0
6	0	0	1	1	1	1	0	0	0		0	0	0	0	1	0	0	0	0
7	0	0	0	0	1	0	0	0	0		0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0

iii)

	A	B	C	D	E	F	G	H	I		J		K	L	M	N	O	P	Q	R	S
1	0	0	0	0	0	0	0	0	0				0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0				0	0	0	0	1	1	0	0	0
3	0	0	0	0	1	1	0	0	0				0	0	1	1	1	1	1	0	0
4	0	0	1	1	1	1	1	0	0				0	1	1	1	1	1	1	1	0
5	0	0	0	1	1	1	1	0	0				0	0	1	1	1	1	1	1	0
6	0	0	1	1	1	1	0	0	0				0	1	1	1	1	1	1	0	0
7	0	0	0	0	1	0	0	0	0				0	0	1	1	1	1	0	0	0
8	0	0	0	0	0	0	0	0	0				0	0	0	0	1	0	0	0	0
9	0	0	0	0	0	0	0	0	0				0	0	0	0	0	0	0	0	0
10																					
11													0	0	0	0	0	0	0	0	0
12													0	0	0	0	0	0	0	0	0
13													0	0	0	0	1	1	0	0	0
14													0	0	1	1	1	1	1	0	0
15													0	0	0	1	1	1	1	0	0
16													0	0	1	1	1	1	0	0	0
17													0	0	0	0	1	0	0	0	0
18													0	0	0	0	0	0	0	0	0
19													0	0	0	0	0	0	0	0	0

iv)

|	A	B	C	D	E	F	G	H	I		J		K	L	M	N	O	P	Q	R	S
1	0	0	0	0	0	0	0	0	0				0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0				0	0	0	0	0	0	0	0	0
3	0	0	0	0	1	1	0	0	0				0	0	0	0	0	0	0	0	0
4	0	0	1	1	1	1	1	0	0				0	0	0	0	1	1	0	0	0
5	0	0	0	1	1	1	1	0	0				0	0	0	0	1	1	0	0	0
6	0	0	1	1	1	1	0	0	0				0	0	0	0	1	0	0	0	0
7	0	0	0	0	1	0	0	0	0				0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0				0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0				0	0	0	0	0	0	0	0	0
10																					
11													0	0	0	0	0	0	0	0	0
12													0	0	0	0	0	0	0	0	0
13													0	0	0	0	1	1	0	0	0
14													0	0	0	1	1	1	1	0	0
15													0	0	0	1	1	1	1	0	0
16													0	0	0	1	1	1	0	0	0
17													0	0	0	0	1	0	0	0	0
18													0	0	0	0	0	0	0	0	0
19													0	0	0	0	0	0	0	0	0

2. A) i) You use opening since we want to do erosion followed by dilation. Erosion is good for removing thin lines and then we can dilation what is left. The erosion will remove the diagonal and vertical lines and then the dilation will expand the horizontal line so all other lines would be gone.
ii) We would use a line structuring element. This way we can use a horizontal line-shape structuring element to detect the lines we want.
iii) This depends on the width of the horizontal lines. We can take the minimum width of the horizontal lines and then make the structuring element slightly larger so that we capture the entire width of the horizontal lines and not remove some parts of horizontal lines.

b) i) Refer to code below. Our shape is line and our degrees is 0 because we want to detect horizontal lines. Then we selected 55 experimentally where it was enough length to show the shortest horizontal line and was long enough that the width of other lines did not appear.

ii)   iii)
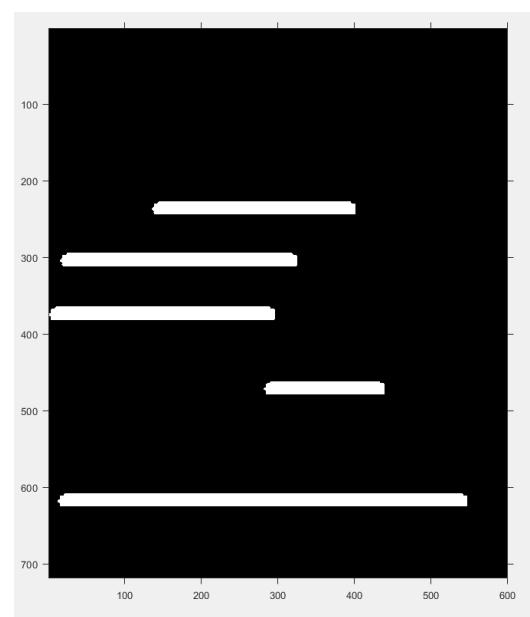
iii) For this one we just increases the length paramter to 110 so it got rid of the shortest line which is the one at the top and the length was enough to kept the rest.

iv) Now we change the degrees part to 90 since we want to find vertical lines. We also set the length to 30 as that is sufficient for being longer than width of other lines and include the shortest vertical line we have.
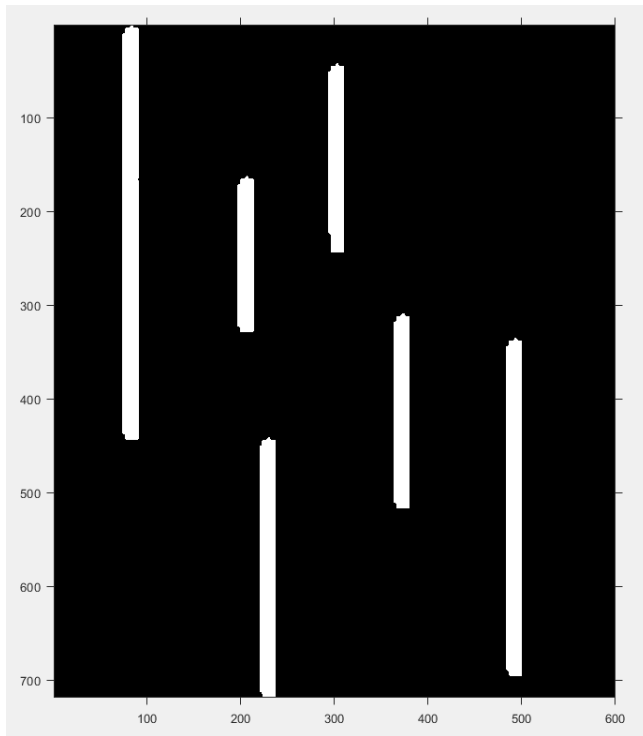
v)
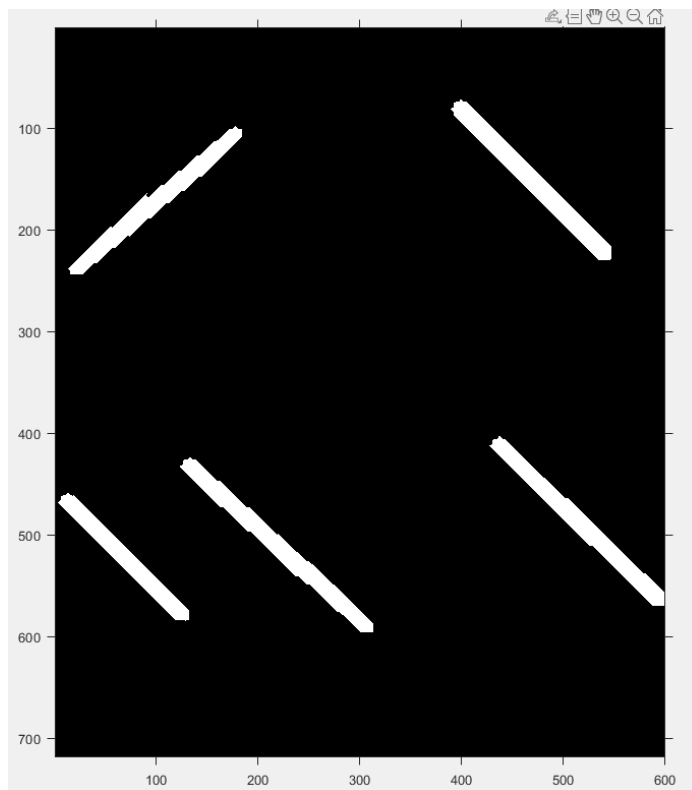
For this one we needed to use two structuring elements so that we can get diagonal lines of both 45 degrees and 135 degrees (180 – 45). We apply the open operation with both structuring elements and then combine the two. 55 was chosen as the length for both as it included the shortest diagonal line and was larger than the width that any other lines may contain.

```matlab
im = imread('asn4.tif');
se = strel('line', 55, 0);
open = imopen(im, se);
figure
imshow(open);

se = strel('line', 110, 0);
open = imopen(im, se);
figure
imshow(open);

se = strel('line', 30, 90);
open = imopen(im, se);
figure
imshow(open);

se1 = strel('line', 55, 45);
se2 = strel('line', 55, 135);
open1 = imopen(im, se1);
open2 = imopen(im, se2);
figure
imshow(max(open1, open2));
```