Q1

XOR is a function which isn't linear.
A single layer perceptron isn't able to classify ~~the~~ non-linear func.
Therefore, my introducing multiple layers ~~of~~ of perceptrons, (MLP) we remove the linearity of the ~~func~~ model, helping us to classify non linear func.

Q2

A linear relation would be of type : $\hat{y}_1 = \beta_0 + \beta_1 x$

Multiple layers of same type would result in:

$$\hat{y} = (\beta_0 + \beta_0' + \beta_0'') + (\beta_1 + \beta_1' + \beta_1'')x$$

$$= \beta_0 + \beta_1 x + \beta_0' + \beta_1' x + \cdots$$

$$= \hat{y}_1 + \hat{y}_2 + \cdots$$

↳ This is result of just addition of multiple linear ~~story~~ func.

∴ linear layers stacked on linear layers remain linear.

In deep neural networks, ~~extensive~~ chain rule is used extensively to calculate error (Backprop.)

Eg: $\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial a_{30}} \frac{\partial a_{30}}{\partial a_{29}} \cdots \frac{\partial a_1}{\partial w_1}$

Here the partial derivatives are usually ~~of the~~ less than 1, which when multiplied lot of times, result in a very small value, ~~as~~ resulting in gradient shrink.

Sigmoid func' takes the value and scales it down to (0, 1) resulting in even smaller derivative.
Relu is max(0, x) which gives this derivative ~~as~~ as 1, solving the problem of vanishing grad.

Q3.

When transformers use self attention, the tokens ~~are~~ aren't given some number to preserve the original sequence. It doesn't know ~~which~~ what the correct order is. To solve this problem, we use positional encoding.

Absolute PE	Sinusoidal PE
Gives unique vectors to each index	Uses sine, cos func' of ~~a~~ different frequency
Initialized randomly and optimized by backprop.	No training happens. Fixed math formula are used
Can only handle seq. ~~of~~ having length less than an assigned val.	Can be used on any length of sequence

RoPE helps with long-context as it shifts how the model perceive distance from 'absolute slots' to relative rotation.
Unlike ~~previous~~ other methods that add vectors together, RoPE rotates the Query & key vectors.

Q4 Query (Q) : ~~The~~ It ~~tells~~ represents the "current" word looking for information
Key (K) : This represents a "label" for all words in the sequence.
Value (V) : This is the actual info tied to the word.

$\sqrt{d_k}$ is used for scaling as it gives us stability numerically.
When d_k is too large, $Q \cdot K^T$ is very ~~big~~ big. Hence by dividing by $\sqrt{d_k}$ val of

Page No.
Date

$Q \cdot K^T$ is ~~•~~ & around 1.

In attention weight matrix, the diagonal represents ~~the~~ a word attending itself. ~~Since~~ As the word relates most to itself, the diagonal values are highest.

Q5 If we use single attention head, the model can only focus on one type of relationship at a time for a specific word. By using multiple heads our model can attend ~~to~~ to different types of relationships at the same time.

d_{model} : Total size of embedding vector for each word.

h : No. of parallel attention "workers".

d_{head} : Size of vector processed by each individual head

Q6

Greedy decoding can ~~lead~~ lead the model into a "local optima" trap. By choosing a very likely word now, the model may be forced ~~to~~ into highly unlikely words later to finish the sentence.

Beam search is better as it explores multiple branches/choices simultaneously. If one branch starts with a slightly lower-prob. word but leads to a much more logical and high-prob. conclusion, beam search will preserve that path while greedy would've discarded ~~it~~ immediately.

	1st Word		2nd Word
Eg	Word 1 → 0.7 prob	⇒	Word 2 → 0.001 prob.
	Word ~~2~~ 1 → 0.3 prob	⇒	Word 2 → 0.9 prob.

1st path prob = 0.007
2nd path prob = 0.27

Page No.
Date

Q1

a) $d_{head} = \frac{d_{model}}{h} = 64$

b) Parameter per Matrix = $d_{model}^2 = (768)^2$

For 3 matrices: $3\, d_{model}^2 = 3 \cdot (768)^2$

Q2

$\frac{e^{z_i}}{\Sigma e^{z_i}}$

$e^2 = 7.39$

$e^1 = 2.71$

$e^0 = 1$

$\Sigma e^{z_i} = 11.1$

$\sigma_1 = \frac{7.39}{11.1} = 0.6657$

$\sigma_2 = \frac{2.71}{11.1} = 0.244$

$\sigma_3 = \frac{1}{11.1} = 0.09$