Question 1 :

- Logistic regression deals with classification problems, where the output is a probability between 0 and 1 representing the likelihood of a particular class. Mean squared error, on the other hand, is designed for regression tasks where we predict a continuous value. So, MSE isn't well-suited for interpreting the classification outputs of logistic regression.
- Cross entropy loss leads to a convex loss function during optimization with logistic regression. This means there's a single minimum point, making it easier for the model to converge and find the optimal solution using gradient descent.
- When MSE is used with logistic regression, it essentially measures the squared difference between the predicted probabilities and the actual labels. However, this doesn't align well with the probabilistic interpretation of logistic regression, where the goal is to estimate the likelihood of class membership.

Question 2 :

- In case of linear activation functions, we can visualize the entire neural network as one single linear function, with respect to the input. In this case, MSE results in a convex optimization problem. We can see this as follows : As the entire neural network is a linear function, our objective is to find the optimal set of weights for which MSE is minimum. See that this problem is same as linear regression!, and it is a well known fact that MSE results in a convex optimization problem when used as the loss function in linear regression. Hence, we can say MSE results in a convex optimization problem for a binary classification task with a deep neural network equipped with linear activation function.
- For CE, when used for binary classification, it also results in a convex optimization problem. Similar to above argument, here what we are doing is same as logistic regression (applying the softmax function does not change the convexity as it itself is convex and composition of two convex functions is convex. Linear functions are also convex).

Therefore, the correct option is option (c)

Question 3 :

- The MNIST dataset was used. A 2 layer dense, feedforward neural network was trained on this dataset. Each hidden layer had 128 units. ReLu was used as the activation function
- For preprocessing, the input images were normalized. Also, some images were randomly rotated or flipped horizontally with a probability=0.5.
- The network was trained for 10 epochs and then tested on the training set. Batch size of 64 was used. Learning rate and batch sizes were the hyperparameters tuned.
- The best set of hyperparameters were when learning rate was 0.001 and batch size was 64.

Question 4 :

- The SVHN dataset was used. ResNet(18, 50 and 101) and VGG-16 was used to classify the images and theire results were compared.
- Adam optimizer was used along with CrossEntropy as the loss fuction. All the models were trained for 10 epochs.
- 1/4 of the dataset was used for training and rest was used as the test set. Accuracy was used as the performance metric
- ResNet18 had the highest accuracy on test set, followed by VGG16. The accuracy values are :
  - Resnet18 : 0.9020257357625175
  - Resnet50 : 0.893944633529294
  - Resnet101 : 0.8892488579072857

  - VGG16 : 0.9019893344011066

- The ability to efficiently propagate features through skip connections could have made the ResNet family well suited for this task. The higher models Resnet50 and Resnet101 have lower accuracy compared to others, which may be due to overfitting on the training set. VGG16 deep architecture also helped it to perform well on the test set.