Homework 2

Please complete the following questions based on the PyTorch tutorial.
https://pytorch.org/docs/stable/nn.html

1.Design a Linear layer using nn. Linear ().
nn.Linear() is used to implement fully connected layer functions. The feature is in B×D format. B denotes batch_size, D denotes the dimension of expected features in the input. Input feature is 32×512. Please design the corresponding layer to output a tensor with 32×256 dimension and print the dimension of the output. Fill in the blanks below to meet the requirements.

```
>>> input = torch.randn(32, 512)
>>> m = nn.Linear(_________)
>>> output = m(input)
```

2. Design a Convolutional Neural Networks using nn.Conv2d().
nn.Conv2d is used to create two-dimensional convolutional layers, typically employed in Convolutional Neural Networks (CNNs). The feature is in B×C×H×W format. B denotes batch_size, C denotes channel, H denotes height, W denotes width. Input feature is 32×36×512×256.

(a) You use conv with 3×3 kernel size. Stride is 2. Please design the corresponding layer to output a tensor with 32×34×255×127 dimension and print the dimension of the output. Fill in the blanks below to meet the requirements.

```
>>> input = torch.randn(32, 36, 512, 256)
>>> m = nn.Conv2d(_____________________)
>>> output = m(input)
```

(b) You use conv with 2×4 kernel size. Strides are 1 and 2 respectively. Padding is 3 and 1 respectively. Please design the corresponding layer to output a tensor with 32×35×517×128 dimension and print the dimension of the output. Fill in the blanks below to meet the requirements.

```
>>> m = nn.Conv2d(_____________________________________________)
>>> output = m(input)
```

(c) You use conv with 3×5 kernel size. Strides are 3 and 2 respectively. Padding is 3 and 1 respectively. Dilations are 2 and 1 respectively. Group is 2. Input feature is 32×16×512×256. Please design the corresponding layer to output a tensor with 32×32×172×127 dimension and print the dimension of the output. Fill in the blanks below to meet the requirements.

```
>>> input = torch.randn(32, 16, 512, 256)
>>> m = nn.Conv2d(_____________________________________________)
>>> output = m(input)
```

3. Design a RNN using nn.RNN().
nn.RNN() is used to create one layer of a Recurrent Neural Network (RNN) model. The feature is

in L×B×D format. L denotes sequence length, B denotes batch_size, D denotes the dimension of expected features in the input. Input feature is 64×8×32. You want to design a 2 layers RNN. The hidden size is 64. Please design the corresponding layer to output a tensor with 64×8×64 dimension and print the dimension of the output. Fill in the blanks below to meet the requirements.

```
>>> input = torch.randn(64, 8, 32)
>>> rnn = nn.RNN(_____________)
>>> h0 = torch.randn(_____________)
>>> output, hn = rnn(input, h0)
```

4.Design a TransformerEncoderLayer using nn.TransformerEncoderLayer().
nn.TransformerEncoderLayer() is used to create the encoding layers of a Transformer model, which includes multi-head self-attention layers, feed-forward neural network layers and layer normalization. The feature is in B×L×D format. B denotes batch_size, L denotes sequence length, D denotes the dimension of expected features in the input. Input feature is 128×16×512. The head number of Transformer is 8. Please design the corresponding layer to output a tensor with 128×16×512 dimension and print the dimension of the output. Set batch_first is True. Fill in the blanks below to meet the requirements.

```
>>> input = torch.rand(128, 16, 512)
>>> encoder_layer = nn.TransformerEncoderLayer(_____________)
>>> out = encoder_layer(input)
```

5. Calculate the number of parameters of depthwise and 1×1 convolution. Assume the feature dimension is in H×W×C format.

（1）Input feature is 16×16×32. We use depthwise conv with 3×3 kernel size. Calculate the number of parameters in the kernel. Assume there is no bias term.

（2）Input feature is 24×24×64. We use 64 1×1 conv kernels. Calculate the number of parameters in the kernel. Assume there is no bias term.