```
# Program to train a simple CNN model
import tensorflow as tf
from tensorflow.keras import datasets,layers,models
import matplotlib.pyplot as plt
import numpy as np
(X_train,y_train),(X_test,y_test)=datasets.cifar10.load_data()

print('Length of training set = ',(len(X_train)))
print('Length of testing set = ',(len(X_test)))
classes=["airplane","automobile","bird","cat","deer","dog","frog","hors
e","ship","truck"]

def plot_sample(X,y,index):
  plt.figure(figsize=(15,2))
  plt.imshow(X[index])
  print(y[index])
  print(classes[(y[index])])
  plt.xlabel(classes[(y[index])])

  #plt.xlabel(classes[y[index]])


y_train=y_train.reshape(-1)
plot_sample(X_train,y_train,12)

X_train=X_train/255
X_test=X_test/255

classes=["airplane","automobile","bird","cat","deer","dog","frog","hors
e","ship","truck"]
def plot_sample(X,y,index):
  plt.figure(figsize=(15,2))
  plt.imshow(X[index])
  print(y[index])
  print(classes[(y[index])])
  plt.xlabel(classes[(y[index])])
  #plt.xlabel(classes[(y[index])])
  #plt.xlabel(classes[y[index]])
print(classes)
y_train=y_train.reshape(-1)
plot_sample(X_train,y_train,10)

X_train=X_train/255
X_test=X_test/255

cnn=models.Sequential([
    layers.Conv2D(filters=32,kernel_size=(3,3),activation='relu',input_
shape=(32,32,3)),
    #layers.Conv2D(filters=64,kernel_size=(3,3),activation='relu'),
```

```
    layers.MaxPool2D((2,2)),
    layers.Flatten(),
    #layers.Dense(64,activation="relu"),
    layers.Dense(10,activation="softmax")
])

cnn.summary()
cnn.compile(optimizer='adam',loss='sparse_categorical_crossentropy',met
rics=['accuracy'])
cnn.fit(X_train,y_train,epochs=1)

cnn.evaluate(X_test,y_test)
```

OUTPUT

```
Length of training set =  50000
Length of testing set =  10000
7
horse
['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse',
'ship', 'truck']
4
deer
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_2 (Conv2D)           (None, 30, 30, 32)        896

 max_pooling2d_2 (MaxPoolin  (None, 15, 15, 32)        0
 g2D)

 flatten_2 (Flatten)         (None, 7200)              0

 dense_2 (Dense)             (None, 10)                72010

=================================================================
Total params: 72906 (284.79 KB)
Trainable params: 72906 (284.79 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
1563/1563 [==============================] - 32s 20ms/step - loss: 2.1432 -
accuracy: 0.2310
313/313 [==============================] - 3s 8ms/step - loss: 2.0457 -
accuracy: 0.2814
313/313 [==============================] - 2s 8ms/step
Classification Report                precision    recall  f1-score   support

           0       0.34      0.46      0.39      1000
           1       0.28      0.28      0.28      1000
           2       0.38      0.02      0.04      1000
           3       0.25      0.03      0.05      1000
           4       0.20      0.32      0.25      1000
           5       0.28      0.37      0.32      1000
```

```
              6       0.26      0.44      0.33      1000
              7       0.28      0.13      0.17      1000
              8       0.37      0.26      0.31      1000
              9       0.30      0.50      0.37      1000

       accuracy                           0.28     10000
      macro avg       0.29      0.28      0.25     10000
   weighted avg       0.29      0.28      0.25     10000
```