In [1]:

```
#assignment-07
#D.prudhvi sai
```

In [64]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [2]:

```
dataset=pd.read_csv('Fish.csv')
```

In [4]:

```
dataset
```

Out[4]:

| | Species | Weight | Length1 | Length2 | Length3 | Height | Width |
|---|---|---|---|---|---|---|---|
| **0** | Bream | 242.0 | 23.2 | 25.4 | 30.0 | 11.5200 | 4.0200 |
| **1** | Bream | 290.0 | 24.0 | 26.3 | 31.2 | 12.4800 | 4.3056 |
| **2** | Bream | 340.0 | 23.9 | 26.5 | 31.1 | 12.3778 | 4.6961 |
| **3** | Bream | 363.0 | 26.3 | 29.0 | 33.5 | 12.7300 | 4.4555 |
| **4** | Bream | 430.0 | 26.5 | 29.0 | 34.0 | 12.4440 | 5.1340 |
| **...** | ... | ... | ... | ... | ... | ... | ... |
| **154** | Smelt | 12.2 | 11.5 | 12.2 | 13.4 | 2.0904 | 1.3936 |
| **155** | Smelt | 13.4 | 11.7 | 12.4 | 13.5 | 2.4300 | 1.2690 |
| **156** | Smelt | 12.2 | 12.1 | 13.0 | 13.8 | 2.2770 | 1.2558 |
| **157** | Smelt | 19.7 | 13.2 | 14.3 | 15.2 | 2.8728 | 2.0672 |
| **158** | Smelt | 19.9 | 13.8 | 15.0 | 16.2 | 2.9322 | 1.8792 |

159 rows × 7 columns

In [5]:

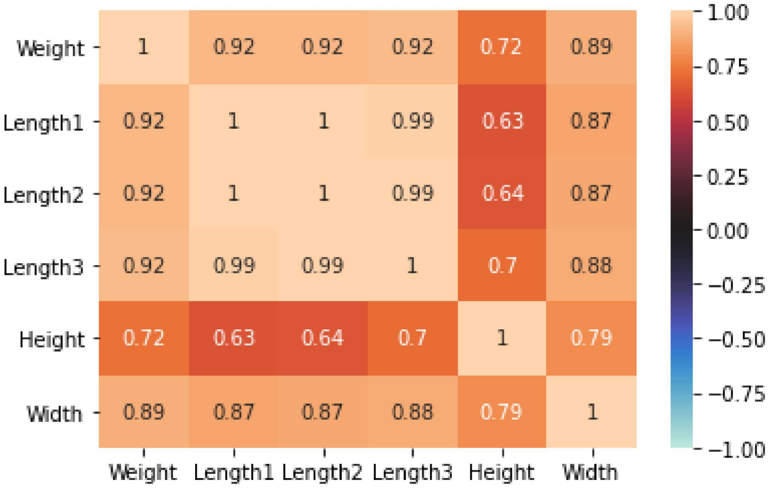
```
dataset.isnull().any()
```

Out[5]:

```
Species     False
Weight      False
Length1     False
Length2     False
Length3     False
Height      False
Width       False
dtype: bool
```

In [6]:
```
import seaborn as sns
sns.heatmap(dataset.corr(),annot=True,vmin=-1,vmax=1,center=0)
```

Out[6]:
```
<AxesSubplot:>
```



In [8]:
```
dataset.head()
```

Out[8]:

| | Species | Weight | Length1 | Length2 | Length3 | Height | Width |
|---|---|---|---|---|---|---|---|
| **0** | Bream | 242.0 | 23.2 | 25.4 | 30.0 | 11.5200 | 4.0200 |
| **1** | Bream | 290.0 | 24.0 | 26.3 | 31.2 | 12.4800 | 4.3056 |
| **2** | Bream | 340.0 | 23.9 | 26.5 | 31.1 | 12.3778 | 4.6961 |
| **3** | Bream | 363.0 | 26.3 | 29.0 | 33.5 | 12.7300 | 4.4555 |
| **4** | Bream | 430.0 | 26.5 | 29.0 | 34.0 | 12.4440 | 5.1340 |

In [9]:
```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
```

In [10]:
```
dataset['Species']=le.fit_transform(dataset['Species'])
```

In [11]:
```
dataset.head()
```

Out[11]:

| | Species | Weight | Length1 | Length2 | Length3 | Height | Width |
|---|---|---|---|---|---|---|---|
| **0** | 0 | 242.0 | 23.2 | 25.4 | 30.0 | 11.5200 | 4.0200 |
| **1** | 0 | 290.0 | 24.0 | 26.3 | 31.2 | 12.4800 | 4.3056 |
| **2** | 0 | 340.0 | 23.9 | 26.5 | 31.1 | 12.3778 | 4.6961 |
| **3** | 0 | 363.0 | 26.3 | 29.0 | 33.5 | 12.7300 | 4.4555 |
| **4** | 0 | 430.0 | 26.5 | 29.0 | 34.0 | 12.4440 | 5.1340 |

In [12]:
```
x=dataset.iloc[:,1:7]
y=dataset.iloc[:,0:1]
```

In [13]:
```
x
```

Out[13]:

| | Weight | Length1 | Length2 | Length3 | Height | Width |
|---|---|---|---|---|---|---|
| **0** | 242.0 | 23.2 | 25.4 | 30.0 | 11.5200 | 4.0200 |
| **1** | 290.0 | 24.0 | 26.3 | 31.2 | 12.4800 | 4.3056 |
| **2** | 340.0 | 23.9 | 26.5 | 31.1 | 12.3778 | 4.6961 |
| **3** | 363.0 | 26.3 | 29.0 | 33.5 | 12.7300 | 4.4555 |
| **4** | 430.0 | 26.5 | 29.0 | 34.0 | 12.4440 | 5.1340 |
| **...** | ... | ... | ... | ... | ... | ... |
| **154** | 12.2 | 11.5 | 12.2 | 13.4 | 2.0904 | 1.3936 |
| **155** | 13.4 | 11.7 | 12.4 | 13.5 | 2.4300 | 1.2690 |
| **156** | 12.2 | 12.1 | 13.0 | 13.8 | 2.2770 | 1.2558 |
| **157** | 19.7 | 13.2 | 14.3 | 15.2 | 2.8728 | 2.0672 |
| **158** | 19.9 | 13.8 | 15.0 | 16.2 | 2.9322 | 1.8792 |

159 rows × 6 columns

In [14]:
```
x.ndim
```

Out[14]:
```
2
```

In [15]:
```
x.shape
```

Out[15]:
```
(159, 6)
```

In [16]:
```
y.shape
```

Out[16]:
```
(159, 1)
```

In [19]:

```
x=dataset.iloc[:,1:7].values
y=dataset.iloc[:,0:1].values
```

In [18]:

```
from sklearn.preprocessing import OneHotEncoder
one=OneHotEncoder()
```

In [23]:

```
y=one.fit_transform(y[:,0:1])
```

In [41]:

```
y = y.toarray()
```

In [42]:

```
y
```

Out[42]:

```
array([[1., 0., 0., ..., 0., 0., 0.],
       [1., 0., 0., ..., 0., 0., 0.],
       [1., 0., 0., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 0., 1., 0.],
       [0., 0., 0., ..., 0., 1., 0.],
       [0., 0., 0., ..., 0., 1., 0.]])
```

In [43]:

```
y.shape
```

Out[43]:

```
(159, 7)
```

In [44]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

In [45]:

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.fit_transform(x_test)
```

In [46]:

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
```

In [47]:

```
categorical = Sequential()
```

In [48]:

```
categorical.add(Dense(units = 6 , kernel_initializer = "random_uniform",activatio
```

In [49]:

```
categorical.add(Dropout(0.2))
```

In [50]:

```
categorical.add(Dense(units = 12 , kernel_initializer = "random_uniform",activati
```

In [51]:

```
categorical.add(Dropout(0.2))
```

In [52]:
```
categorical.add(Dense(units = 12 , kernel_initializer = "random_uniform",activati
```

In [53]:
```
categorical.add(Dropout(0.2))
```

In [54]:
```
categorical.add(Dense(units = 12 , kernel_initializer = "random_uniform",activati
```

In [55]:
```
categorical.add(Dense(units = 7 , kernel_initializer = "random_uniform",activatic
```

In [56]:
```
categorical.compile(optimizer = "rmsprop",loss="categorical_crossentropy" ,metric
```

In [57]:
```
history = categorical.fit(x_train,y_train, batch_size =128,epochs = 1000, validat
```

```
Epoch 1/1000
1/1 [==============================] - 1s 857ms/step - loss: 1.9459 - accurac
y: 0.2126 - val_loss: 1.9446 - val_accuracy: 0.1875
Epoch 2/1000
1/1 [==============================] - 0s 43ms/step - loss: 1.9437 - accurac
y: 0.2283 - val_loss: 1.9436 - val_accuracy: 0.1875
Epoch 3/1000
1/1 [==============================] - 0s 54ms/step - loss: 1.9420 - accurac
y: 0.2283 - val_loss: 1.9428 - val_accuracy: 0.1875
Epoch 4/1000
1/1 [==============================] - 0s 40ms/step - loss: 1.9406 - accurac
y: 0.2283 - val_loss: 1.9420 - val_accuracy: 0.1875
Epoch 5/1000
1/1 [==============================] - 0s 45ms/step - loss: 1.9392 - accurac
y: 0.2283 - val_loss: 1.9412 - val_accuracy: 0.1875
Epoch 6/1000
1/1 [==============================] - 0s 47ms/step - loss: 1.9379 - accurac
y: 0.2283 - val_loss: 1.9405 - val_accuracy: 0.1875
Epoch 7/1000
```

In [58]:
```
yp = categorical.predict(sc.transform([[120,17.5,19,21.3,8.3922,2.9181]]))
```

In [59]:
```
yp
```

Out[59]:
```
array([[0.02690255, 0.09684834, 0.28754437, 0.03541575, 0.23103298,
        0.27790752, 0.04434848]], dtype=float32)
```

In [60]:
```
yp = categorical.predict(sc.transform([[20,17.5,42.3,134.3,8.3922,0.9181]]))
```

In [61]:
```
yp = np.argmax(yp)#[0,0,0,0,1,0]
```

In [62]:
```
index = ["bearm","parkki","perch","pike","roach","smelt","whitefih"]
```

In [63]:
```
prediction = index[yp]
```

In [ ]: