In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
from sklearn import metrics
from sklearn.metrics import accuracy_score
from sklearn.naive_bayes import MultinomialNB
from sklearn.multiclass import OneVsRestClassifier
from pandas.plotting import scatter_matrix
from sklearn.neighbors import KNeighborsClassifier
```

In [4]:

```
data=pd.read_csv("Resume_DataSet (3).csv")
data['cleaned_resume']= ' '
data
```

Out[4]:

| | Category | Resume | cleaned_resume |
|---|---|---|---|
| **0** | Data Science | Skills * Programming Languages: Python (pandas... | |
| **1** | Data Science | Education Details \r\nMay 2013 to May 2017 B.E... | |
| **2** | Data Science | Areas of Interest Deep Learning, Control Syste... | |
| **3** | Data Science | Skills â□¢ R â□¢ Python â□¢ SAP HANA â□¢ Table... | |
| **4** | Data Science | Education Details \r\n MCA YMCAUST, Faridab... | |
| **...** | ... | ... | ... |
| **957** | Testing | Computer Skills: â□¢ Proficient in MS office (... | |
| **958** | Testing | â□□ Willingness to accept the challenges. â□□ ... | |
| **959** | Testing | PERSONAL SKILLS â□¢ Quick learner, â□¢ Eagerne... | |
| **960** | Testing | COMPUTER SKILLS & SOFTWARE KNOWLEDGE MS-Power ... | |
| **961** | Testing | Skill Set OS Windows XP/7/8/8.1/10 Database MY... | |

962 rows × 3 columns

In [5]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 962 entries, 0 to 961
Data columns (total 3 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Category        962 non-null    object
 1   Resume          962 non-null    object
 2   cleaned_resume  962 non-null    object
dtypes: object(3)
memory usage: 22.7+ KB
```

In [6]:

```
print("The Different Categories in the Resume are:")
print("\n")
print(data['Category'].unique())
```

```
The Different Categories in the Resume are:


['Data Science' 'HR' 'Advocate' 'Arts' 'Web Designing'
 'Mechanical Engineer' 'Sales' 'Health and fitness' 'Civil Engineer'
 'Java Developer' 'Business Analyst' 'SAP Developer' 'Automation Testing'
 'Electrical Engineering' 'Operations Manager' 'Python Developer'
 'DevOps Engineer' 'Network Security Engineer' 'PMO' 'Database' 'Hadoop'
 'ETL Developer' 'DotNet Developer' 'Blockchain' 'Testing']
```

In [7]:

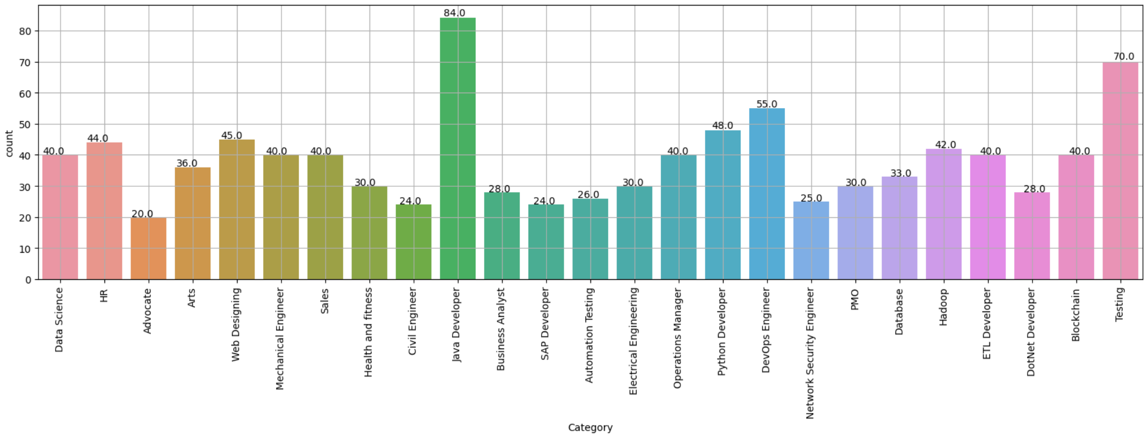
```
print("The Different Categories in the Resume and the number of records belongin
print("\n")
print(data['Category'].value_counts())
```

```
The Different Categories in the Resume and the number of records belonging to e
ach category are as follows:


Java Developer               84
Testing                      70
DevOps Engineer              55
Python Developer             48
Web Designing                45
HR                           44
Hadoop                       42
Blockchain                   40
ETL Developer                40
Operations Manager           40
Data Science                 40
Sales                        40
Mechanical Engineer          40
Arts                         36
Database                     33
Electrical Engineering       30
Health and fitness           30
PMO                          30
Business Analyst             28
DotNet Developer             28
Automation Testing           26
Network Security Engineer    25
SAP Developer                24
Civil Engineer               24
Advocate                     20
Name: Category, dtype: int64
```

In [8]:

```
import seaborn as sns
plt.figure(figsize=(20,5))
plt.xticks(rotation=90)
ax=sns.countplot(x="Category", data=data)
for p in ax.patches:
    ax.annotate(str(p.get_height()), (p.get_x() * 1.01 , p.get_height() * 1.01))
plt.grid()
```



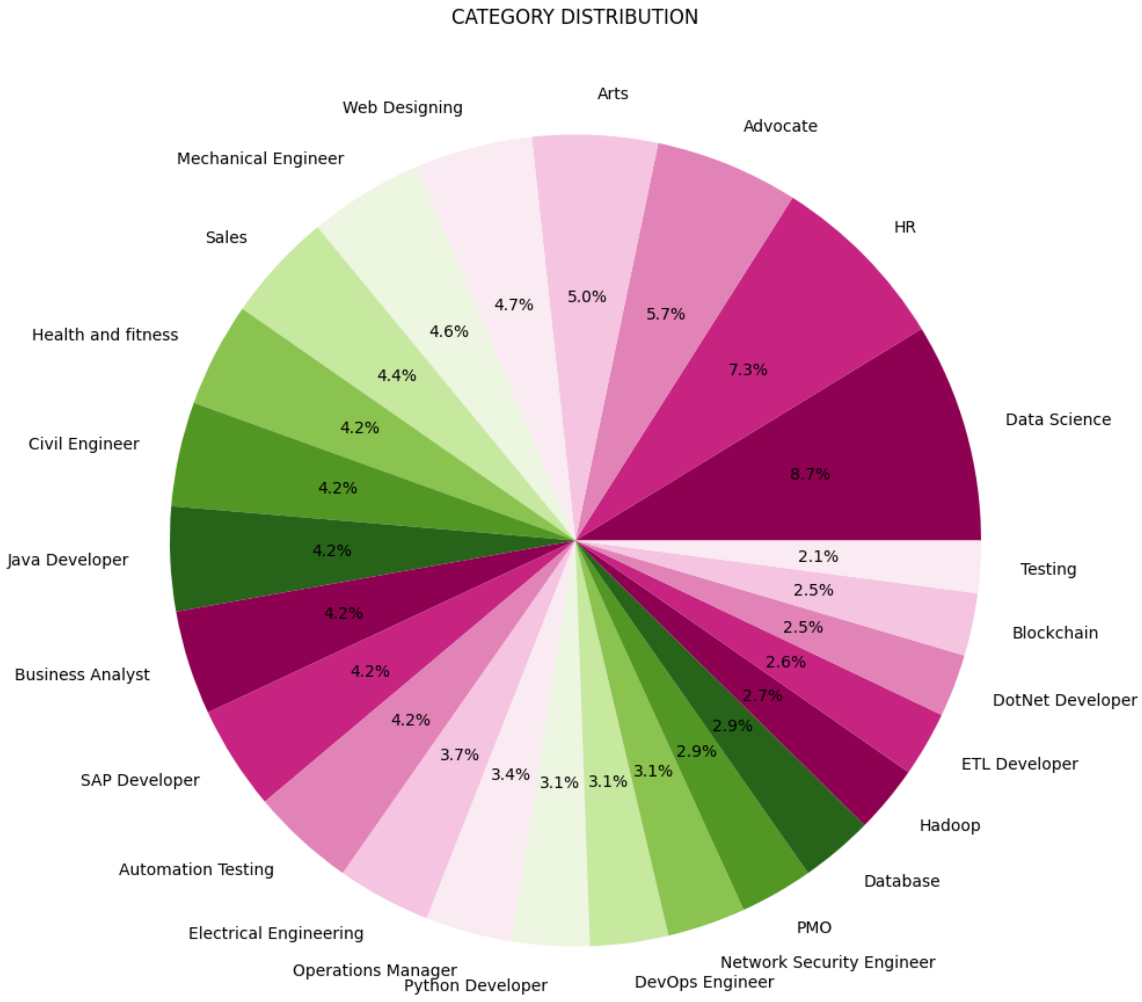
In [9]:

```python
from matplotlib.gridspec import GridSpec
count=data['Category'].value_counts()
labels=data['Category'].unique()

plt.figure(1, figsize=(25,25))
the_grid= GridSpec(2,2)

cmap=plt.get_cmap('PiYG')
colors= [cmap(i) for i in np.linspace(0,1,10)]
plt.subplot(the_grid[0,1], aspect=1, title='CATEGORY DISTRIBUTION')

plt.pie(count, labels=labels, autopct='%1.1f%%', colors=colors)
plt.show()
```



In [10]:
```
import re
def cleanResume(resumeText):
    resumeText=re.sub('http\S+\s*', ' ', resumeText)
    resumeText=re.sub('RT|cc', ' ', resumeText)
    resumeText=re.sub('#\S+', ' ', resumeText)
    resumeText=re.sub('@\S+', ' ', resumeText)
    resumeText=re.sub('[%s]' %re.escape("""!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~"""),
    resumeText=re.sub(r'[^\x00-\x7f]', r' ', resumeText)
    resumeText=re.sub('\s+', ' ', resumeText)
    return resumeText

data['cleaned_resume']=data.Resume.apply(lambda x: cleanResume(x))
```

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

Out[11]:

| | Category | Resume | cleaned_resume |
|---|---|---|---|
| **0** | Data Science | Skills * Programming Languages: Python (pandas... | Skills Programming Languages Python pandas num... |
| **1** | Data Science | Education Details \r\nMay 2013 to May 2017 B.E... | Education Details May 2013 to May 2017 B E UIT... |
| **2** | Data Science | Areas of Interest Deep Learning, Control Syste... | Areas of Interest Deep Learning Control System... |
| **3** | Data Science | Skills â□¢ R â□¢ Python â□¢ SAP HANA â□¢ Table... | Skills R Python SAP HANA Tableau SAP HANA SQL ... |
| **4** | Data Science | Education Details \r\n MCA YMCAUST, Faridab... | Education Details MCA YMCAUST Faridabad Haryan... |

In [13]:
```
import nltk
from nltk.corpus import stopwords
import string
from wordcloud import WordCloud

oneSetOfStopWords = set(stopwords.words('english')+['``',"''"])
totalWords =[]
Sentences = data['Resume'].values
cleanedSentences = ""
for i in range(0,160):
    cleanedText = cleanResume(Sentences[i])
    cleanedSentences += cleanedText
    requiredWords = nltk.word_tokenize(cleanedText)
    for word in requiredWords:
        if word not in oneSetOfStopWords and word not in string.punctuation:
            totalWords.append(word)

wordfreqdist = nltk.FreqDist(totalWords)
mostcommon = wordfreqdist.most_common(50)
print(mostcommon)

wc = WordCloud().generate(cleanedSentences)
plt.figure(figsize=(15,15))
plt.imshow(wc, interpolation='bilinear')
plt.axis("off")
plt.show()
```

```
[('Details', 484), ('Exprience', 446), ('months', 376), ('company', 330), ('des
cription', 310), ('1', 290), ('year', 232), ('January', 216), ('Less', 204),
('Data', 200), ('data', 192), ('Skill', 166), ('Maharashtra', 166), ('6', 164),
('Python', 156), ('Science', 154), ('I', 146), ('Education', 142), ('College',
140), ('The', 126), ('project', 126), ('like', 126), ('Project', 124), ('Learni
ng', 116), ('India', 114), ('Machine', 112), ('University', 112), ('Web', 106),
('using', 104), ('monthsCompany', 102), ('B', 98), ('C', 98), ('SQL', 96), ('ti
me', 92), ('learning', 90), ('Mumbai', 90), ('Pune', 90), ('Arts', 90), ('A', 8
4), ('application', 84), ('Engineering', 78), ('24', 76), ('various', 76), ('So
ftware', 76), ('Responsibilities', 76), ('Nagpur', 76), ('development', 74),
('Management', 74), ('projects', 74), ('Technologies', 72)]
```

Education Details
Machine Learning
Company Details
Maharashtra State
Team size
Project Description
Exprience Less
HR Skill
year monthsCompany
monthsCompany Details
Python Exprience
Operating System
Exprience months
Deep Learning
experience
SAP HANA
CSS BOOTSTRAP
Exprience monthsCompany
client
year months
Technical Environment
Council description
India
Details company
Skill Details
project
Pvt Ltd
HTML CSS
State Board
company British
year
Details January
Nagpur Maharashtra
British Council
Data Science
developed
Project Name

In [14]:

```python
from sklearn.preprocessing import LabelEncoder

var_mod = ['Category']
le = LabelEncoder()
for i in var_mod:
    data[i] = le.fit_transform(data[i])
```

In [15]:

```python
data.head()
```

Out[15]:

| | Category | Resume | cleaned_resume |
|---|---|---|---|
| **0** | 6 | Skills * Programming Languages: Python (pandas... | Skills Programming Languages Python pandas num... |
| **1** | 6 | Education Details \r\nMay 2013 to May 2017 B.E... | Education Details May 2013 to May 2017 B E UIT... |
| **2** | 6 | Areas of Interest Deep Learning, Control Syste... | Areas of Interest Deep Learning Control System... |
| **3** | 6 | Skills â¢ R â¢ Python â¢ SAP HANA â¢ Table... | Skills R Python SAP HANA Tableau SAP HANA SQL ... |
| **4** | 6 | Education Details \r\n MCA YMCAUST, Faridab... | Education Details MCA YMCAUST Faridabad Haryan... |

In [16]:

```python
data.Category.value_counts()
```

Out[16]:

```
15    84
23    70
8     55
20    48
24    45
12    44
13    42
3     40
10    40
18    40
6     40
22    40
16    40
1     36
7     33
11    30
14    30
19    30
4     28
9     28
2     26
17    25
21    24
5     24
0     20
Name: Category, dtype: int64
```

In [17]:

```
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from scipy.sparse import hstack

requiredText = data['cleaned_resume'].values
requiredTarget = data['Category'].values

word_vectorizer = TfidfVectorizer(
    sublinear_tf=True,
    stop_words='english',
    max_features=1500)
word_vectorizer.fit(requiredText)
WordFeatures = word_vectorizer.transform(requiredText)

print ("Feature completed .....")

x_train,x_test,y_train,y_test = train_test_split(WordFeatures,requiredTarget,rar
print(x_train.shape)
print(x_test.shape)
```

```
Feature completed .....
(769, 1500)
(193, 1500)
```

In [19]:

```
from sklearn.linear_model import LogisticRegression

lr=LogisticRegression()
lr.fit(x_train,y_train)
pred=lr.predict(x_test)

from sklearn.metrics import confusion_matrix
cnf_matrix=confusion_matrix(y_test,pred)
acc=accuracy_score(y_test,pred)
```

```
print("Confusion Matrix")
print(cnf_matrix)
print("\n")
print("Accuracy of Logistic Regression is:", acc)

print("\n Classification report for %s:\n%s\n" % (lr, metrics.classification_rep
Confusion Matrix
[[ 3  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  3  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  4  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  9  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  6  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  5  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  9  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  7  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0 10  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  9  0  0  0  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  8  0  0  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  9  0  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  5  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  9  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  7  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 19  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  3  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  4  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  5  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  6  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 11  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  4  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 13  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 15
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
   8]]


Accuracy of Logistic Regression is: 0.9896373056994818

 Classification report for LogisticRegression():
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         3
           1       1.00      1.00      1.00         3
           2       1.00      0.80      0.89         5
           3       1.00      1.00      1.00         9
           4       1.00      1.00      1.00         6
           5       1.00      1.00      1.00         5
           6       1.00      1.00      1.00         9
           7       1.00      1.00      1.00         7
           8       1.00      0.91      0.95        11
           9       1.00      1.00      1.00         9
          10       1.00      1.00      1.00         8
          11       0.90      1.00      0.95         9
          12       1.00      1.00      1.00         5
          13       1.00      1.00      1.00         9
          14       1.00      1.00      1.00         7
          15       0.95      1.00      0.97        19
          16       1.00      1.00      1.00         3
          17       1.00      1.00      1.00         4
          18       1.00      1.00      1.00         5
          19       1.00      1.00      1.00         6
          20       1.00      1.00      1.00        11
          21       1.00      1.00      1.00         4
          22       1.00      1.00      1.00        13
          23       1.00      1.00      1.00        15
          24       1.00      1.00      1.00         8

    accuracy                           0.99       193
   macro avg       0.99      0.99      0.99       193
weighted avg       0.99      0.99      0.99       193
```

In [20]:

```
clf = OneVsRestClassifier(KNeighborsClassifier())
clf.fit(x_train, y_train)
prediction = clf.predict(x_test)

from sklearn.metrics import confusion_matrix
cnf_matrix=confusion_matrix(y_test,pred)
acc=accuracy_score(y_test,prediction)

print("Confusion Matrix")
print(cnf_matrix)
print("\n")
print("Accuracy of KNeighbors Classifier is:", acc)

print("\n Classification report for classifier %s:\n%s\n" % (clf, metrics.classi
Confusion Matrix
[[ 3  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  3  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  4  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  9  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  6  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  5  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  9  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  7  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0 10  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  9  0  0  0  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  8  0  0  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  9  0  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  5  0  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  9  0  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  7  0  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 19  0  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  3  0  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  4  0  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  5  0  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  6  0  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 11  0  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  4  0  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 13  0
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 15
   0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
   8]]


Accuracy of KNeighbors Classifier is: 0.9896373056994818

 Classification report for classifier OneVsRestClassifier(estimator=KNeighborsC
lassifier()):
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         3
           1       1.00      1.00      1.00         3
           2       1.00      0.80      0.89         5
           3       1.00      1.00      1.00         9
           4       1.00      1.00      1.00         6
           5       0.83      1.00      0.91         5
           6       1.00      1.00      1.00         9
           7       1.00      1.00      1.00         7
           8       1.00      0.91      0.95        11
           9       1.00      1.00      1.00         9
          10       1.00      1.00      1.00         8
          11       0.90      1.00      0.95         9
          12       1.00      1.00      1.00         5
          13       1.00      1.00      1.00         9
          14       1.00      1.00      1.00         7
          15       1.00      1.00      1.00        19
          16       1.00      1.00      1.00         3
          17       1.00      1.00      1.00         4
          18       1.00      1.00      1.00         5
          19       1.00      1.00      1.00         6
          20       1.00      1.00      1.00        11
          21       1.00      1.00      1.00         4
          22       1.00      1.00      1.00        13
          23       1.00      1.00      1.00        15
          24       1.00      1.00      1.00         8

    accuracy                           0.99       193
   macro avg       0.99      0.99      0.99       193
weighted avg       0.99      0.99      0.99       193
```

```
In [ ]:
```