100 Python Interview Questions & Answers for Data Analyst and Data Science

Programming Assignment Helper

1. Question: Reverse a string

Answer:

```
s = 'hello'
print(s[::-1])
```

2. Question: Check if a number is prime

Answer:

```
n = 7
print(all(n % i != 0 for i in range(2, int(n**0.5) + 1)))
```

3. Question: Find factorial using recursion

Answer:

```
def fact(n):
    return 1 if n == 0 else n * fact(n-1)
print(fact(5))
```

4. Question: Check palindrome string

Answer:

```
s = 'madam'
print(s == s[::-1])
```

5. Question: Find largest number in a list

Answer:

```
nums = [1, 5, 2]
print(max(nums))
```

6. Question: Swap two variables

Answer:

```
a, b = 5, 10
a, b = b, a
print(a, b)
```

7. Question: Count vowels in a string

Answer:

```
s = 'hello'
print(sum(1 for ch in s if ch in 'aeiou'))
```

8. Question: Check Armstrong number

Answer:

```
n = 153
print(sum(int(d) ** 3 for d in str(n)) == n)
```

9. Question: Generate Fibonacci series

Answer:

```
a, b = 0, 1
for _ in range(5):
    print(a)
    a, b = b, a + b
```

10. Question: Find GCD of two numbers

Answer:

```
import math
print(math.gcd(12, 15))
```

11. Question: Check anagram

Answer:

```
s1, s2 = 'listen', 'silent'
print(sorted(s1) == sorted(s2))
```

12. Question: Reverse words in a sentence

Answer:

```
s = 'hello world'
    print(' '.join(s.split()[::-1]))
```

13. Question: Find duplicates in a list

Answer:

```
nums = [1, 2, 2, 3]
print([x for x in set(nums) if nums.count(x) > 1])
```

14. Question: Find second largest number in a list

Answer:

```
nums = [1, 2, 3, 4]
print(sorted(set(nums))[-2])
```

15. Question: Check if string contains only digits

Answer:

```
print('123'.isdigit())
```

16. Question: Flatten nested list

Answer:

```
nested = [[1, 2], [3, 4]]
flat = [x for sub in nested for x in sub]
print(flat)
```

17. Question: Sort dictionary by values

Answer:

```
d = {'a': 2, 'b': 1}
print(dict(sorted(d.items(), key=lambda x: x[1])))
```

18. Question: Count words in a string

Answer:

```
s = 'this is test'
print(len(s.split()))
```

19. Question: Check leap year

Answer:

```
y = 2024
print(y % 4 == 0 and (y % 100 != 0 or y % 400 == 0))
```

20. Question: Merge two dictionaries

Answer:

```
d1 = {'a': 1}
d2 = {'b': 2}
d1.update(d2)
print(d1)
```

21. Question: Group words by first letter

Answer:

```
from collections import defaultdict
words =
["apple","ant","banana","ball","cat","car"]
    grouped = defaultdict(list)
 for word in words:
        grouped[word[0]].append(word)
    print(dict(grouped))
```

22. Question: Find all pairs in list whose sum equals target

Answer:

```
nums = [1,2,3,4,5,6,7]
    target = 8
```

```
pairs = [(a,b) for i,a in enumerate(nums) for b in nums[i+1:] if a+b==target]
    print(pairs)
```

23. Question: Convert list of tuples into dictionary

Answer:

```
    pairs = [("a",1),("b",2),("c",3)]
    print(dict(pairs))
```

24. Question: Find top 3 most frequent elements in a list

Answer:

```
from collections import Counter
nums = [1,2,2,3,3,3,4,4,4,4,5]
print(Counter(nums).most_common(3))
```

25. Question: Transpose a matrix

Answer:

```
matrix = [[1,2,3],[4,5,6],[7,8,9]]

transpose = [[row[i] for row in matrix] for i
in range(len(matrix[0]))]

    print(transpose)
```

26. Question: Implement stack using Python list

Answer:

```
    stack = []

    stack.append(10)

    stack.append(20)

    print(stack.pop())
```

27. Question: Implement queue using collections.deque

Answer: from collections

import deque queue = deque()

queue.append(10)

queue.append(20)

print(queue.popleft())

28. Question: Generate random numbers without random module

Answer:

import time

def pseudo_random(seed=1):

    seed = (seed*9301+49297) % 233280

    return seed/233280.0

print(pseudo_random(int(time.time())))

29. Question: Convert string to title case

Answer:

```
text = "data science with python"
print(text .title())
```

30. Question: Remove punctuation from string

Answer:

```
import string
text = "Hello, World! Data@Science."
clean = "".join(c for c in text if c not in string.punctuation)
print(clean)
```

31. Question: Load CSV in pandas and display first 10 rows

Answer:

```
import pandas as pd
```

```
df = pd.read_csv("data.csv")
print(df.head(10))
```

32. Question: Select rows where column value > 100

Answer:

```
print(df[df["Sales"] > 100])
```

33. Question: Drop rows with missing values

Answer:

```
print(df.dropna())
```

34. Question: Fill missing values with column mean

Answer:

```
df["Sales"].fillna(df["Sales"].mean(), inplace=True)
```

35. Question: Merge two DataFrames on a column

Answer:

```
df1 = pd.DataFrame({"ID":[1,2],"Name":["A","B"]})
df2 = pd.DataFrame({"ID":[1,2],"Age":[25,30]})
merged = pd.merge(df1,df2,on="ID")
print(merged)
```

36. Question: Group data by column and calculate mean

Answer:

```
print(df.groupby("Category")[" Sales"].mean())
```

37. Question: Sort DataFrame by multiple columns

Answer:

print(df.sort_values(by=["Category","Sales"], ascending=[True,False]))

38. Question: Apply custom function to column

Answer:

df["Discounted"] = df["Sales"].apply(lambda x: x*0.9)

39. Question: Find number of unique values in a column

Answer:

print(df["CustomerID"].nunique())

40. Question: Rename multiple columns in DataFrame

Answer:

df.rename(columns= {"Sales":"Total_Sales","Date":"Order_Date"}, inplace=True)

41. Question: Create a NumPy array of zeros

Answer:

```
import numpy as np
    arr = np.zeros((3,3))
    print(arr)
```

42. Question: Create a NumPy array from 1 to 10

Answer:

```
    arr = np.arange(1,11)
    print(arr)
```

43. Question: Reshape 1D NumPy array to 2D

Answer:

```
arr = np.arange(1,7).reshape(2,3)
print(arr)
```

44. Question: Find max and min in NumPy array

Answer:

```
arr = np.array([3,7,1,9,2])
print(arr.max(), arr.min())
```

45. Question: Compute mean, median, std of array

Answer:

```
arr = np.array([1,2,3,4,5,6])
print(arr.mean(), np.median(arr), arr.std())
```

46. Question: Slice first 3 elements of array

Answer:

```
arr = np.array([10,20,30,40,50])
print(arr[:3])
```

47. Question: Find index of max in NumPy array

Answer:

```
arr = np.array([4,8,2,9,6])
print(arr.argmax())
```

48. Question: Create diagonal matrix in NumPy

Answer:

```
arr = np.diag([1,2,3])
print(arr)
```

49. Question: Multiply two NumPy arrays element-wise

Answer:

```
a = np.array([1,2,3])
b = np.array([4,5,6])
print(a*b)
```

50. Question: Perform matrix multiplication

Answer:

```
a = np.array([[1,2],[3,4]])
b = np.array([[5,6],[7,8]])
print(np.dot(a,b))
```

51. Question: Create random NumPy array of shape 2x3

Answer:

```
arr = np.random.rand(2,3)
print(arr)
```

52. Question: Normalize a NumPy array

Answer:

```
arr = np.array([10,20,30])
norm = (arr-arr.min())/(arr.max()-arr.min())
print(norm)
```

53. Question: Get unique values from NumPy array

Answer:

```
arr = np.array([1,2,2,3,3,3,4])
print(np.unique(arr))
```

54. Question: Replace negative values with zero

Answer:

```
arr = np.array([-2,5,-7,8])
arr[arr < 0] = 0
print(arr)
```

55. Question: Check for NaN in NumPy array

Answer:

```
arr = np.array([1,np.nan,3])
print(np.isnan(arr))
```

56. Question: Convert NumPy array to Python list

Answer:

```
arr = np.array([1,2,3])
print(arr.tolist())
```

57. Question: Filter rows in pandas with multiple conditions

Answer:

```
df = pd.DataFrame({"A":[1,2,3,4],"B":[10,20,30,40]})
    print(df[(df["A"] > 2) & (df["B"] < 40)])
```

58. Question: Create pivot table in pandas

Answer:

```
df = pd.DataFrame({"Category":["A","A","B","B"],"Sales":[100,200,300,400]})
    pivot = df.pivot_table(values="Sales", index="Category", aggfunc="sum")
    print(pivot)
```

59. Question: Get correlation matrix

Answer:

```
    print(df.corr())
```

60. Question: Save DataFrame to CSV

Answer:

```
df.to_csv("output.csv", index=False)
```

61. Question: Select specific columns

Answer:

```
df = pd.DataFrame({"A":[1,2,3],"B":
[4,5,6],"C":[7,8,9]})
print(df[["A","C"]])
```

62. Question: Sort DataFrame by column descending

Answer:

```
print(df.sort_values("B", ascending=False))
```

63. Question: Rename columns

Answer:

```
df = df.rename(columns=
{"A":"Col1","B":"Col2"})
```

```
print(df)
```

64. Question: Drop rows with missing values

Answer:

```
df = pd.DataFrame({"A":[1,None,3],"B":
[4,5,None]})
```

```
print(df.dropna())
```

65. Question: Fill missing values with mean

Answer:

```
df["A"].fillna(df["A"].mean(), inplace=True)
```

```
print(df)
```

66. Question: Convert column to datetime

Answer:

```
df = pd.DataFrame({"Date":["2023-01-01","2023-02-01"]})
df["Date"] = pd.to_datetime(df["Date"])
print(df.dtypes)
```

67. Question: Extract year and month from datetime

Answer:

```
df["Year"] = df["Date"].dt.year
df["Month"] = df["Date"].dt.month
print(df)
```

68. Question: Remove duplicate rows

Answer:

```
df = pd.DataFrame({"A":[1,2,2,3],"B":[4,5,5,6]})
```

```
print(df.drop_duplicates())
```

69. Question: Group by column and sum

Answer:

```
df = pd.DataFrame({"Category":
["A","A","B"],"Sales":[100,200,300]})

print(df.groupby("Category")
[" Sales"].sum())
```

70. Question: Apply custom function to column

Answer:

```
df = pd.DataFrame({"A":[1,2,3]})

df["Square"] = df["A"].apply(lambda x:x**2)

print(df)
```

71. Question: Merge two DataFrames on key

Answer:

```
df1 = pd.DataFrame({"ID":[1,2],"Name":
["A","B"]})
df2 = pd.DataFrame({"ID":[1,2],"Age":
[25,30]})
print(pd.merge(df1,df2,on="ID"))
```

72. Question: Concatenate two DataFrames vertically

Answer:

```
df1 = pd.DataFrame({"A":[1,2]})
df2 = pd.DataFrame({"A":[3,4]})
  print(pd.concat([df1,df2]))
```

73. Question: Find top 2 rows by column

Answer:

```
df = pd.DataFrame({"A":[10,40,30,20]})
```

```
print(df.nlargest(2,"A"))
```

74. Question: Find bottom 2 rows by column

Answer:

```
print(df.nsmallest(2,"A"))
```

75. Question: Count missing values

Answer:

```
df = pd.DataFrame({"A":[1,None,3],"B":
[None,5,6]})
 print(df.isnull().sum())
```

76. Question: Replace specific value

Answer:

```
df = pd.DataFrame({"A":[1,2,2,3]})
    df["A"].replace(2, 99, inplace=True)
    print(df)
```

77. Question: Convert categorical to dummy variables

Answer:

```
df = pd.DataFrame({"Fruit": ["Apple","Banana","Apple"]})
print(pd.get_dummies(df, columns= ["Fruit"]))
```

78. Question: Calculate cumulative sum

Answer:

```
df = pd.DataFrame({"Sales": [100,200,300]})
df["Cumulative"] = df["Sales"].cumsum()
print(df)
```

79. Question: Calculate percentage of total

Answer:

```
df["Percentage"] = df["Sales"]/df["Sales"].sum()*100
print(df)
```

80. Question: Detect outliers using IQR

Answer:

```
Q1 = df["Sales"].quantile(0.25)
Q3 = df["Sales"].quantile(0.75)
IQR = Q3 - Q1
outliers = df[(df["Sales"] < Q1 - 1.5*IQR) | (df["Sales"] > Q3 + 1.5*IQR)]
print(outliers)
```

81. Question: Standardize a numeric column using z-score

Answer:

```
df["Zscore"] = (df["Sales"] - df["Sales"].mean()) / df["Sales"].std()
```

```
print(df )
```

82. Question: Normalize values between 0 and 1

Answer:

```
df["Normalized"] = (df["Sales"]-df[" Sales"].min())/(df[" Sales"].max()-df[" Sales"].min())

print(df)
```

83. Question: Split dataset into train and test

Answer:

```
from sklearn.model_selection import train_test_split

X = df[["Sales"]]

y = [0,1,0,1]

X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.2,random_stat
```

e=42)

print(X_train,y_train)

84. Question: Train simple linear regression

Answer:

```
from sklearn.linear_model import
LinearRegression
    X = np.array([[1],[2],[3],[4]])
    y = np.array([2,4,6,8])
 model = LinearRegression()
 model.fit(X,y)
 print(model.coef_, model.intercept_)
```

85. Question: Make predictions

Answer:

```
pred = model.predict([[5]])
    print(pred)
```

86. Question: Evaluate regression using R2 score

Answer:

```
from sklearn.metrics import r2_score
  y_true = [2,4,6,8]
y_pred = model.predict(X)
  print(r2_score(y_true,y_pred))
```

87. Question: Train logistic regression

Answer:

```
from sklearn.linear_model import LogisticRegression
  X = np.array([[1],[2],[3],[4]])
  y = np.array([0,0,1,1])
  clf = LogisticRegression()
  clf.fit(X,y)
```

```
print(clf.predict([[2.5]]))
```

88. Question: Create confusion matrix

Answer:

```
from sklearn.metrics import confusion_matrix

y_true = [0,0,1,1]

y_pred = [0,1,1,1]

print(confusion_matrix(y_true,y_pred))
```

89. Question: Calculate accuracy, precision, recall, f1-score

Answer:

```
from sklearn.metrics import classification_report

print(classification_report(y_true,y_pred))
```

90. Question: Perform k-fold cross-validation

Answer:

```
from sklearn.model_selection import cross_val_score

scores = cross_val_score(clf,X,y,cv=3)

print(scores.mean())
```

91. Question: Train decision tree classifier

Answer:

```
from sklearn.tree import DecisionTreeClassifier

clf = DecisionTreeClassifier()

clf.fit(X,y)

print(clf.predict([[2]]))
```

92. Question: Train random forest classifier

Answer:

```
from sklearn.ensemble import RandomForestClassifier
clf = RandomForestClassifier()
clf.fit(X,y)
print(clf.predict([[3]]))
```

93. Question: Train support vector machine classifier

Answer:

```
from sklearn.svm import SVC
clf = SVC()
clf.fit(X,y)
print(clf.predict([[2.5]]))
```

94. Question: Scale features using StandardScaler

Answer:

```
from sklearn.preprocessing import StandardScaler
   scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
   print(X_scaled)
```

95. Question: Encode categorical using LabelEncoder

Answer:

```
from sklearn.preprocessing import LabelEncoder
fruits = ["Apple","Banana","Apple","Orange"]
   encoder = LabelEncoder()
encoded = encoder.fit_transform(fruits)
   print(encoded)
```

96. Question: One-hot

encode categorical feature

Answer:

```
from sklearn.preprocessing import OneHotEncoder
encoder = OneHotEncoder(sparse=False)
data = np.array(fruits).reshape(-1,1)
encoded = encoder.fit_transform(data)
print(encoded)
```

97. Question: Reduce dimensions using PCA

Answer:

```
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
X_reduced = pca.fit_transform(X_scaled)
```

```
print(X_reduced)
```

98. Question: Save trained model with joblib

Answer:

```
import joblib
joblib.dump(clf,"model.pkl")
```

99. Question: Load trained model with joblib

Answer:

```
model = joblib.load("model.pkl")
print(model.predict([[2]]))
```

100. Question: Create pipeline with scaler and classifier

Answer:

```
from sklearn.pipeline import Pipeline
```

```
    pipeline = Pipeline([("scaler",
StandardScaler()),("clf",
LogisticRegression())])

    pipeline.fit(X,y)

    print(pipeline.predict([[2.5]]))
```

Programming Assignment Helper

share to help others

save to refer to later

Follow me for daily posts on strategy, mindset, and professional development.

WhatsApp number : +8801772432396 or +8801750264761

WhatsApp Account : https://wa.me/+8801750264761
https://wa.me/+8801772432396

Email : programmingassignmentsservices@gmail.com
academiccourseworkservices@gmail.com

Our work portfolio Link : https://rumble.com/c/c-7708281